DETCHEM™
User Manual

Version 2.5

http://www.detchem.com

Olaf Deutschmann *, Steffen Tischer, Chrys Correa,
Daniel Chatterjee, Stefan Kleditzsch, Vinod Janardhanan,
Nikolay Mladenov, Hoang Duc Minh, Hüseyin Karadeniz,
Matthias Hettel

c/o Olaf Deutschmann

STZ 240 Simulation of Reactive Flows

Germany

October 22, 2014

*Tel.: +49 7247 9542387, E-Mail: mail@detchem.com

Preface

DETCHEM (**DET**ailed **CHEMistry**) is a package of **FORTRAN** routines for the modeling of gaseous reactive flows including reactions on solid surfaces using elementary-step like reaction mechanisms as well as global reaction mechanisms. It is also designed to couple **DET** ailed **CHEM**istry with Computational Fluid Dynamics(CFD) programs. This approach allows the simulation of both the flow and chemistry using models based on the molecular processes. The **DETCHEM** software package consists of several computer programs.

DETCHEM can also be coupled to the open source CFD software package **OpenFOAM** as described in this manual

With version 2.1 **DETCHEM** software package is restructured, uniform input format has been introduced and several tools have been added. **DETCHEM** has been applied to simulations in the following fields: catalytic reactors, catalytic combustion, automotive catalytic converters, micro-reactors, micro-burners, solid-oxid fuel cells, chemical vapor deposition etc.

The **FORTRAN** routines used in the **DETCHEM**-package have been developed by O. Deutschmann, S. Tischer, V. Janardhanan, C. Correa, D. Chatterjee, S. Kleditzsch, N. Mladenov, and H. D. Minh. **DETCHEM** applies the DAE-solvers LIMEX (developed by P. Deuflhard et al., Berlin, Germany) and DAESOL (developed by H.-G. Bock et al., Heidelberg, Germany) LIMEX and DAESOL are not part of **DETCHEM**.

Fruitful discussions with many colleagues helped to develop the code, in particular we would like to thank J. Warnatz(University of Heidelberg), H. Zhu and R. J. Kee (Colorado School of Mines).

No guarantee of faultless performance of the software is given. The DAE solvers applied in the software, the mathematical libraries, and compilers needed are not part of DETCHEM. Reaction mechanisms are not part of the software. A selection of reaction mechanisms is available and can be downloaded from www.detchem.com for free. The software is distributed by the STZ 240 Reaktive Strömungen, Heidelberg, Germany, within the Steinbeis-Stiftung. For further information, please contact mail@detchem.com.

Contents

1 Getting Started **9**
- 1.1 Introduction . . . 9
- 1.2 Program structure . . . 10
- 1.3 Program Capabilities . . . 10

2 Fundamentals of DETCHEM **11**
- 2.1 Species . . . 11
  - 2.1.1 General . . . 11
  - 2.1.2 Gas-phase species . . . 11
  - 2.1.3 Third-body species . . . 14
  - 2.1.4 Surface species . . . 15
- 2.2 Chemical Reactions . . . 15
  - 2.2.1 General . . . 15
  - 2.2.2 Gas-phase reactions . . . 17
  - 2.2.3 Pressure-dependent gas-phase reactions . . . 17
  - 2.2.4 Surface Reactions . . . 18
- 2.3 Modeling of transport and surface reactions in washcoats . . . 19
  - 2.3.1 Diffusion and reaction in porous catalysts . . . 19
  - 2.3.2 Detailed washcoat model . . . 19
  - 2.3.3 Effectiveness factor approach . . . 19
  - 2.3.4 Effective diffusion coefficients . . . 20

3 Input files **21**
- 3.1 General structure of inp-files . . . 21
  - 3.1.1 Tags . . . 21
  - 3.1.2 Options and Values . . . 22
  - 3.1.3 Commands . . . 22
  - 3.1.4 Comments . . . 23
  - 3.1.5 Units . . . 23
- 3.2 thermdata . . . 24
- 3.3 moldata . . . 25
- 3.4 Gas-phase chemistry mechanism file . . . 25
- 3.5 Surface chemistry mechanism file . . . 26
- 3.6 Species input . . . 28
- 3.7 Mechanism input . . . 29
- 3.8 Surface models . . . 35
  - 3.8.1 **<CHEMSURF>** input . . . 36
  - 3.8.2 Catalytic vs. geometric surface area . . . 36
  - 3.8.3 Washcoat diffusion models . . . 37
  - 3.8.4 Effectiveness factor model . . . 39

4 DETCHEM$^{\text{CHEMINP}}$ **41**
- 4.1 Introduction . . . 41
- 4.2 User Input . . . 41
- 4.3 Output . . . 41
- 4.4 Isotope analysis . . . 41
- 4.5 Running the tool . . . 42

5 DETCHEM^{GASPROBE} 43
5.1 Introduction . . . 43
5.2 User Input . . . 43
5.3 Output . . . 43
5.4 Examples . . . 44
5.5 Setting up a problem . . . 44
5.6 Running the tool . . . 44

6 DETCHEM^{SURFPROBE} 45
6.1 Introduction . . . 45
6.2 User Input . . . 45
6.3 Output . . . 45
6.4 Examples . . . 46
6.5 Setting up a problem . . . 46
6.6 Running the tool . . . 46

7 DETCHEM^{EQUIL} 47
7.1 Introduction . . . 47
7.1.1 Theoretical background . . . 47
7.1.2 Algorithm . . . 48
7.2 User input . . . 48
7.2.1 ***equil.inp*** . . . 48
7.2.2 ***thermdata*** . . . 49
7.3 Output . . . 49
7.3.1 Screen output . . . 49
7.3.2 ***equil.plt*** . . . 50
7.4 Example . . . 50
7.5 Setting up a problem . . . 51

8 DETCHEM^{BATCH} 53
8.1 Introduction . . . 53
8.1.1 Governing equations . . . 53
8.1.2 Solution . . . 54
8.2 User Input . . . 54
8.3 Output . . . 57
8.3.1 Screen output . . . 57
8.3.2 ***batch_***.plt*** . . . 57
8.3.3 ***batch_sens_***.txt*** . . . 58
8.4 Examples . . . 58
8.5 Setting up a problem . . . 58
8.6 Running the tool . . . 58

9 DETCHEM^{CSTR} 59
9.1 Introduction . . . 59
9.1.1 Governing equations . . . 59
9.1.2 Solution . . . 59
9.2 User Input . . . 59
9.3 Output . . . 60
9.3.1 Screen output . . . 60
9.3.2 ***cstr***.plt*** . . . 61
9.3.3 Examples . . . 61
9.4 Setting up a problem . . . 61
9.5 Running the tool . . . 61

10 DETCHEM^{STAG} **63**
10.1 Introduction . . . 63
10.2 Governing Equations . . . 63
10.2.1 Gas-phase equations . . . 63
10.2.2 Surface equations . . . 64
10.2.3 Boundary conditions . . . 64
10.3 Solution . . . 65
10.4 User Input . . . 65
10.5 Output . . . 67
10.5.1 Screen output . . . 67
10.5.2 gas_***.dat . . . 67
10.5.3 surf_***.dat . . . 67
10.5.4 wcoat_***.dat . . . 67
10.5.5 wsurf_***.dat . . . 68
10.5.6 wcstag_***.dat . . . 68
10.6 Examples . . . 68
10.7 Setting up a problem . . . 68
10.8 Running the tool . . . 68

11 DETCHEM^{PLUG} **69**
11.1 Introduction . . . 69
11.1.1 Governing Equations . . . 69
11.1.2 Mass and Heat transfer coefficients . . . 70
11.1.3 Solution . . . 70
11.2 User Input . . . 70
11.2.1 The **<GEOMETRY>** tag . . . 72
11.2.2 The **<SECTION>** tag . . . 72
11.2.3 The **<SOLVER>** tag . . . 72
11.2.4 The **<SUMMARY>** and **<SUM>** tag . . . 72
11.3 Output . . . 72
11.3.1 Screen output . . . 72
11.3.2 ***outg***.plt*** . . . 73
11.3.3 ***outs***.plt*** . . . 73
11.3.4 ***trans***.plt*** . . . 73
11.3.5 ***summary.dat*** . . . 73
11.4 Examples . . . 74
11.5 Setting up a problem . . . 74
11.6 Running the tool . . . 74

12 DETCHEM^{PACKEDBED} **75**
12.1 Introduction . . . 75
12.1.1 Governing Equations . . . 75
12.2 Solution . . . 76
12.3 User Input . . . 76
12.3.1 The **<SECTION>** tag . . . 77
12.3.2 The **<WALL>** tag . . . 77
12.3.3 The **<SOLVER>** tag . . . 77
12.3.4 The **<TOLERANCE>** tag . . . 77
12.3.5 The **<OUTPUT>** tag . . . 77
12.4 Output . . . 77
12.4.1 Screen output . . . 77
12.4.2 ***outg***.plt*** . . . 78
12.4.3 ***outs***.plt*** . . . 78
12.5 Examples . . . 78
12.6 Setting up a problem . . . 79
12.7 Running the tool . . . 79

13 DETCHEM$^{\text{CHANNEL}}$ **81**
13.1 Introduction . . . 81
13.1.1 The boundary layer approach . . . 81
13.1.2 Governing Equations . . . 81
13.1.3 Solution . . . 82
13.2 User Input . . . 83
13.2.1 The **<BASICS>** tag . . . 84
13.2.2 The **<SECTION>** tag . . . 84
13.2.3 The **<SOLVER>** tag . . . 84
13.2.4 The **<TOLERANCE>** tag . . . 85
13.2.5 The **<OUTPUT>** tag . . . 85
13.2.6 The **<INLET>** tag . . . 86
13.2.7 The **<TPROFILE>** tag . . . 86
13.3 Output . . . 87
13.3.1 Screen output . . . 87
13.3.2 ***outg***.plt*** . . . 87
13.3.3 ***outs***.plt*** . . . 87
13.3.4 ***outflux***.plt*** . . . 87
13.3.5 ***summary_out.dat*** . . . 88
13.4 Examples . . . 88
13.5 Setting up a problem . . . 88
13.6 Running the tool . . . 88
13.7 BLAYER version . . . 89
13.7.1 User Input . . . 89
13.7.2 Output . . . 89

14 DETCHEM$^{\text{GRIDGEN3D}}$ **91**
14.1 Introduction . . . 91
14.2 User Input . . . 91
14.2.1 The **<POLYGON>** tag . . . 92
14.2.2 The **<CIRCUMPOINTS>** tag . . . 92
14.2.3 The **<TECPLOT>** tag . . . 93
14.2.4 The **<LAYER>** tag . . . 93
14.2.5 The **<BORDER>** tag . . . 94
14.3 Output . . . 94
14.3.1 Screen output . . . 94
14.3.2 ***r_grid.inp*** . . . 94
14.3.3 ***r_grid.plt*** . . . 94
14.4 Examples . . . 95
14.5 Running the tool . . . 95

15 DETCHEM$^{\text{RESERVOIR}}$ **97**
15.1 Introduction . . . 97
15.2 User input . . . 98
15.3 Output . . . 101
15.4 Examples . . . 102
15.5 Running the tool . . . 102

16 DETCHEM$^{\text{SOFC}}$ **103**
16.1 Introduction . . . 103
16.2 Physical and chemical fundamentals . . . 103
16.2.1 Heat and mass transport . . . 103
16.3 User Input . . . 104
16.4 Output . . . 109
16.4.1 Examples . . . 112
16.4.2 Setting up a problem . . . 112
16.4.3 Running the tool . . . 112

17 DC4OpenFOAM **113**
17.1 Introduction . . . 113
17.1.1 Modelling of Complex Flow with Detailed Chemistry . . . 113
17.1.2 Simulation Tools **DUO** and **DC4OpenFOAM** . . . 113
17.1.3 How **DUO** and **DC4OpenFOAM** work . . . 114
17.2 Programming of **DC4OpenFOAM** . . . 116
17.2.1 OpenFOAM . . . 116
17.2.2 Code Files . . . 117
17.3 Usage of **DC4OpenFOAM** . . . 117
17.3.1 Installation and Test . . . 117
17.3.2 Preprocessing . . . 119
17.3.3 Control of **DC4OpenFOAM** . . . 123
17.3.4 Start of a Calculation with **DC4OpenFOAM** . . . 124
17.3.5 Solution Data from OpenFOAM . . . 124
17.4 Appendix . . . 124
17.4.1 Example of an Input File ***properties.inp*** . . . 124

18 DUO **127**
18.1 Introduction . . . 127
18.1.1 Background . . . 127
18.1.2 Modelling of a Monolithic Reformer . . . 127
18.1.3 Simulation Tools **DUO** and **DC4OpenFOAM** . . . 128
18.1.4 How **DUO** and **DC4OpenFOAM** work . . . 128
18.2 Programming of **DUO** . . . 131
18.2.1 OpenFOAM . . . 131
18.2.2 Code Files . . . 132
18.3 Usage of **DUO** . . . 133
18.3.1 General . . . 133
18.3.2 Installation and Test . . . 133
18.3.3 Preprocessing . . . 135
18.3.4 Control of **DUO** . . . 139
18.3.5 Start of a Calculation with **DUO** . . . 142
18.3.6 Data Exchange . . . 142
18.3.7 Data Output . . . 143
18.4 Appendix . . . 143
18.4.1 Example of a Keyword File ***couplingParameters*** . . . 143
18.4.2 Example of an Input File ***plug.inp*** . . . 146
18.4.3 Example of an Input File ***channel.inp*** . . . 147

19 DETCHEM Parameters **151**

20 Installation guide **153**
20.1 Extract data . . . 153
20.2 Run Make . . . 154

Chapter 1

Getting Started

This chapter provides an introduction to DETCHEM, an explanation of its structure and capabilites. The detailed explanation of the models are presented in the subsequent chapters.

1.1 Introduction

Chemical reaction engineering and combustion processes are very often characterized by complex interactions between transport and chemical kinetics. The chemistry may include gas phase as well as surface reactions.

The simulation of fluid flows, which includes detailed schemes for surface and gas phase chemistry, has recently received considerable attention due to the availability of faster computers, the development of new numerical algorithms and the establishment of elementary reaction mechanisms. A key problem still remaining is the stiffness of the governing equations because of the different time scales introduced by chemical reactions including adsorption and desorption. Therefore simulations often use a simplified model, either for the flow field or for the chemistry. This simplification can be risky if there are strong interactions between flow and chemistry. In this case, extrapolations of the achieved results to conditions, which are different from those used for the model validation, are not reliable.

The DETCHEM software package consequently applies detailed models for the description of the chemical reactions and transport processes. It has been designed for a better understanding of the interactions between transport and chemistry and can assist in reactor and process development and optimization.

The basic models of DETCHEM are

- DETCHEM$^{\mathbf{GASPROBE}}$
- DETCHEM$^{\mathbf{SURFPROBE}}$
- DETCHEM$^{\mathbf{EQUIL}}$
- DETCHEM$^{\mathbf{BATCH}}$
- DETCHEM$^{\mathbf{CSTR}}$
- DETCHEM$^{\mathbf{STAG}}$
- DETCHEM$^{\mathbf{PLUG}}$
- DETCHEM$^{\mathbf{CHANNEL}}$
- DETCHEM$^{\mathbf{GRIDGEN3D}}$
- DETCHEM$^{\mathbf{MONOLITH}}$
- DETCHEM$^{\mathbf{RESERVOIR}}$
- DETCHEM$^{\mathbf{SOFC}}$
- DC4OpenFOAM
- DUO

1.2 Program structure

All the DETCHEM models are built up on the library modules ***lib_detchem***, ***lib_utils*** and ***lib_input***. ***lib_detchem*** comprises all library routines for calculating the chemical reaction source terms, while ***lib_utils*** comprises the auxiliary routines for the standard solvers and ***lib_input*** consists of all the routines for reading the newly designed input format of version 2.0.

1.3 Program Capabilities

DETCHEM has the following modeling capabilities

DETCHEM$^{\text{GASPROBE}}$ is a computational tool that calculates homogeneous chemical reaction rates and species thermodynamic and transport data at given conditions of temperature, pressure and composition.

DETCHEM$^{\text{SURFPROBE}}$ is a computational tool that calculates the surface coverages and fluxes at catalytic surfaces at given conditions of temperature, pressure and composition.

DETCHEM$^{\text{EQUIL}}$ is a computational tool to calculate equilibrium gas-phase compositions on the basis of thermodynamical data.

DETCHEM$^{\text{BATCH}}$ is a tool that simulates the temporal variations of temperature and concentrations of gas-phase species as a result of homogeneous gas-phase and/or heterogeneous surface chemical reactions.

DETCHEM$^{\text{CSTR}}$ is a tool that simulates the time dependent variations of temperature and concentrations of gas-phase species as a result of homogeneous gas-phase and/or heterogeneous surface chemical reactions in a continuous stirred tank reactor.

DETCHEM$^{\text{STAG}}$ is a tool that simulates the one dimensional profile of a reacting flow in a stagnation point flow configuration.

DETCHEM$^{\text{PLUG}}$ is a computational tool that simulates one dimensional reacting flows, with and without mass and heat transfer coefficients.

DETCHEM$^{\text{CHANNEL}}$ is a computational tool that simulates parabolic two-dimensional reacting flows in straight channels and annulus using boundary-layer approach.

DETCHEM$^{\text{MONOLITH}}$ is a computational tool that simulates the two dimensional as well as the three-dimensional heat field in monoliths. The reactive flows inside the monolith channels are simulated by the tools DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$.

DETCHEM$^{\text{GRIDGEN3D}}$ is a pre-processor that can be used to set up the computational grid for three-dimensional monolith simulations with DETCHEM$^{\text{MONOLITH}}$.

DETCHEM$^{\text{RESERVOIR}}$ is a computational tool that simulates the transient concentrations within a storage catalyst. It serves as a transient one-dimensional wrapper model for single channel simulations with DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$.

DETCHEM$^{\text{SOFC}}$ is a computational tool that simulates the two-dimensional transient fields within a Solid Oxide Fuel Cell (SOFC).

DC4OpenFOAM is a library which provides transport properties, gas-phase and surface chemistry to OpenFOAM to calculate an arbitrary number of coupled fluid and solid regions in three dimensions.

DUO is an interface to OpenFOAM to calculate monoliths coupled with 3D flow regions upstream. The flow, the gas-phase and surface chemistry in the tubes are handled in 1D or 2D optionally.

Chapter 2

Fundamentals of DETCHEM

The core of all DETCHEM applications is a library of routines that is called to calculate species data and chemical source terms. The species properties include thermodynamic and transport parameters. Whereas for the chemistry reaction schemes based on elementary and global reaction mechanism are used.

2.1 Species

2.1.1 General

The smallest model level is an entity called species. A species represents identical oder similar objects (molecules, adsobats, etc.) of a system in the thermodynamic limit. Microscopic properties of single particles are considered only indirectly in the derivation of transport coefficients.

The fundamental property of a species is its concentration c. Furthermore, if the species is formed by real particles, a molar mass M can also be defined.

The set of all Species is called a thermodynamic system $\mathbf{S}$. Depending on the kind of interactions among the species, it is convenient to consider various subsets of the system.

2.1.2 Gas-phase species

The gas phase $\mathbf{S}_\mathrm{g}$ is formed by the gas-phase species (atomic gases, molecules, radicals, etc.). The particles consist of one or more atoms. Their number, composition, electronic state etc. are characteristics for each species.

For the composition of the gas, we shall consider the following physical properties: total concentration

$$c_\mathrm{total} = \sum_{i \in \mathbf{S}_\mathrm{g}} c_i \tag{2.1}$$

and mean molar mass

$$\bar{M} = \frac{\sum_{i \in \mathbf{S}_\mathrm{g}} c_i M_i}{c_\mathrm{total}} \quad . \tag{2.2}$$

Instead of looking at the composition of a mixture with respect to the total volume, the mixture is quatifyed with respect to moles or mass. Then we can define the mole fraction

$$X_i = \frac{c_i}{c_{total}} \tag{2.3}$$

and the mass fraction

$$Y_i = \frac{M_i}{\bar{M}} X_i \quad . \tag{2.4}$$

Under the assumption of the ideal gas law, we can write the density ρ for given pressure p and temperature T as

$$\rho = \frac{p\bar{M}}{RT} \quad . \tag{2.5}$$

R is the gas constant.

Thermodynamic parameters

For most systems we assume an almost constant static pressure. Therefore, the thermodynamic potential of choice is the enthalpy H. The enthalpy density per mole h is a specific property of each species. When we also include the enthalpy of formation $\Delta_R \overline{H}_f^0$ in the definition, then the total enthalpy is conserved in a reacting system under isobaric conditions.

The heat capacity C gives the relation between heat δQ and a temperature change

$$C = \frac{\delta Q}{\mathrm{d}T} \quad . \tag{2.6}$$

The specific heat capacity at constant pressure c_p is then connected to the enthalpy density by:

$$c_p = \frac{\mathrm{d}h}{\mathrm{d}T} \quad . \tag{2.7}$$

Under the assumption of the ideal gas law, the heat capacity at constant volume is given by

$$c_V = c_p - R \quad . \tag{2.8}$$

Translations, rotations and vibrations contribute to c_V, which can be written as follows:

$$c_V = c_{V,tr} + c_{V,rot} + c_{V,vib} \quad , \tag{2.9}$$

$$c_{V,tr} = \frac{3}{2} R \quad , \tag{2.10}$$

$$c_{V,rot} = \frac{f}{2} R \quad . \tag{2.11}$$

f is the number of excited rotational degrees of freedom (0 for atoms, 2 for linear molecules, 3 for others).

Another variable of interest is the entropy S. According to the second law of thermodymanics it can be written as:

$$\mathrm{d}S = \frac{\delta Q}{T} \quad . \tag{2.12}$$

The thermodynamic properties of species i is described by a polynomial fit of fourth order to the specific heat at constant pressure:

$$C_{pt}^o = R \sum_{n=1}^{5} a_{ni} T^{(n-1)} = R(a_{1i} + a_{2i}T + a_{3i}T^2 + a_{4i}T^3 + a_{5i}T^4) \tag{2.13}$$

Once the specific heat is known, the other thermodynamic properties can be evaluated in terms of the specific heats. The standard state enthalpy is given as

$$H_i^o = \int_{T^o}^{T} C_{pt}^o(T')dT' \tag{2.14}$$

and the standard state entropy

$$S_i(T) = \int_{T^o}^{T} \frac{C_{pt}^o(T')}{T'} dT' \tag{2.15}$$

and with $T^o = 0\text{K}$, ${S_i}^o$ and ${H_i}^o$ can be written as polynomial fits

$$H_i^o = R(a_{1i}T + \frac{a_{2i}}{2}T^2 + \frac{a_{3i}}{3}T^3 + \frac{a_{4i}}{4}T^4 + \frac{a_{5i}}{5}T^5 + a_{6i}) \tag{2.16}$$

The constant of integration a_{6i} is the standard heat of formation at 0K. This constant can be evaluated from the knowledge of standard heat of formation at 298 K

$$S_i^o = R(a_{1i}lnT + a_{2i}T + \frac{a_{3i}}{2}T^2 + \frac{a_{4i}}{3}T^3 + \frac{a_{5i}}{4}T^4 + a_{7i}) \tag{2.17}$$

The constant of integration a_{7i} can be evaluated from the standard state entropy at 298 K.

Diffusion coefficients

The description of the interaction between the particles is based on the model of the Lennard-Jones-(6-12)-potential:

$$E_{\text{L.-J.}}(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \quad , \tag{2.18}$$

where σ is the diameter of the molecule and ε the depth of the potential. The deviations from the model of solid spheres are expressed in terms of the reduced collision integral $\Omega^{(l,s)*}(T^*)$, which is only a function of the reduced temperature

$$T^* = \frac{k_\text{B}T}{\varepsilon} \tag{2.19}$$

(k_B- Boltzmann constant, T - temperature).

The diffusive transport due to a gradient in concentration can be described by the kinetic theory of diluted gases by *Chapman* and *Enskog*. We define the following reduced quantities:

- reduced molar mass

$$M_{ij} = \frac{M_iM_j}{M_i + M_j} \quad , \tag{2.20}$$

- reduced diameter

$$\sigma_{ij} = \frac{1}{2}(\sigma_i + \sigma_j)\xi^{-\frac{1}{6}} \tag{2.21}$$

- reduced potential depth

$$\varepsilon_{ij} = \sqrt{\varepsilon_i\varepsilon_j}\,\xi^2 \quad . \tag{2.22}$$

ξ is a correction factor to account for the interactions between polar (index p) and non-polar particles (index n)

$$\xi_{\text{np}} = 1 + \frac{1}{4}\left(\frac{\alpha_\text{n}}{\sigma_\text{n}^3}\right)\left(\frac{\mu_\text{p}^2}{4\pi\varepsilon_0\varepsilon_\text{p}\sigma_\text{p}^3}\right)\sqrt{\frac{\varepsilon_\text{p}}{\varepsilon_\text{n}}} \quad , \tag{2.23}$$

with α polarizability, μ dipole moment, and ε_0 permeability constant. In all other cases, ξ equal unity.

With these definitions, we can express the binary diffusion coefficient as a function of temperature T and pressure p:

$$D_{ij} = \frac{3}{16}\frac{\sqrt{2\pi\,N_\text{A}k_\text{B}^3T^3/M_{ij}}}{p\pi\sigma_{ij}^2\Omega_{ij}^{(1,1)*}(T_{ij}^*)} \quad , \tag{2.24}$$

(N_A - Avogadro constant).

For simplicity, most calculations use and averaged diffusion coefficient, which can be approximated by:

$$D_{i,\text{M}} = \frac{1 - Y_i}{\sum_{j\in\text{S}_\text{g}\setminus\{i\}}\frac{X_i}{D_{ij}}} \quad . \tag{2.25}$$

Viscosity

The kinetic theory yields a result for the viscosity of a pure species:

$$\eta_i = \frac{5}{16}\frac{\sqrt{\pi M_i k_B T/N_A}}{\pi\sigma_i^2\Omega^{(2,2)*}(T_i^*)} \quad . \tag{2.26}$$

Here, an averaged coefficient of viscosity shall be used, which can be approximated by

$$\eta = \frac{1}{2}\left[\sum_{i\in\mathbf{S}_g} X_i\eta_i + \left(\sum_{i\in\mathbf{S}_g}\frac{X_i}{\eta_i}\right)^{-1}\right] \quad . \tag{2.27}$$

Heat conductivity

Heat conduction and viscosity in gases are caused by transfer of energy and momentum, respectively. Therefore, they are related to each other. However, translational, rotational, and vibrational energy give different contributions :

$$\lambda_i = \frac{\eta_i}{M_i}\left(f_{tr}c_{v,tr} + f_{rot}c_{v,rot} + f_{vib}c_{v,vib}\right) \quad . \tag{2.28}$$

Using the following relations

$$f_{tr} = \frac{5}{2}\left(1 - \frac{2}{\pi}\frac{c_{v,rot}}{c_{v,tr}}\frac{A}{B}\right) \quad , \tag{2.29}$$

$$f_{vib} = \frac{\rho D_{ii}}{\eta_i} \quad , \tag{2.30}$$

$$f_{rot} = f_{vib}\left(1 + \frac{2}{\pi}\frac{A}{B}\right) \quad , \tag{2.31}$$

with

$$A = \frac{5}{2} - f_{vib} \tag{2.32}$$

and

$$B = Z_{rot} + \frac{2}{\pi}\left(\frac{5}{3}\frac{c_{v,rot}}{R} + f_{vib}\right) \quad . \tag{2.33}$$

Z_{rot} is also a characteristic parameter. Once know for a given temperature (298 K), it can be calculated by the proportionality

$$\frac{1}{Z_{rot}(T^*)} \propto 1 + \frac{\pi^{\frac{3}{2}}}{2}T^{*-\frac{1}{2}} + \left(\frac{\pi^2}{4} + 2\right)T^{*-1} + \pi^{\frac{3}{2}}T^{*-\frac{3}{2}} \quad . \tag{2.34}$$

Averaging is handled in analogy to the viscosity by

$$\lambda = \frac{1}{2}\left[\sum_{i\in\mathbf{S}_g} X_i\lambda_i + \left(\sum_{i\in\mathbf{S}_g}\frac{X_i}{\lambda_i}\right)^{-1}\right] \quad . \tag{2.35}$$

2.1.3 Third-body species

In most recombination and decomposition reactions, a third particle is involved in order to guarantee conservation of energy and momentum. Any species of the mixture can take this part, which therefore is denoted by the Symbol M:

$$\mathrm{A + B + M} \rightleftharpoons \mathrm{AB + M} \quad . \tag{2.36}$$

The efficiency of collisions with diffent species is different. In order to summarize all reactions of the same kind, we can define an effective concentration of species M by:

$$c_{\mathrm{M}} \quad = \quad \sum_{i \in \mathbf{S}_{\mathrm{g}}} \alpha_{\mathrm{M},i} c_i \quad , \tag{2.37}$$

where $\alpha_{\mathrm{M},i}$ are the collision efficiencies of the gas-phase species i with respect to a reference species (e.g. Argon).

The set of all third-body species shall be denoted by $\mathbf{S}_{\mathrm{M}}$.

2.1.4 Surface species

In analogy to the gas-phase, the properties of the surface are described in terms of a set of surface species $\mathbf{S}_{\mathrm{s}}$. Every catalytic material and in fact every surface structure has different catalytic properties. Therefore, it might be convenient to divide the set $\mathbf{S}_{\mathrm{s}}$ into subsets $\mathbf{S}_{\mathrm{s}}^j$ ($j = 1, 2, \ldots$); e. g. $\mathbf{S}_{\mathrm{s}} = \mathbf{S}_{\mathrm{Pt}} \cup \mathbf{S}_{\mathrm{Rh}}$.

On each surface type, there is only a limited number of adsorption sites available. An adsorption can be viewed as a reaction of a gas-phase species with an empty site. Thus, empty surface-sites shall be considered as a species.

The total number of surface sites is a conserved quantity. However, some species may occupy more than one site. For convenience we can define the surface coverage fraction

$$\theta_i \quad = \quad \frac{c_i \, \sigma_i}{\Gamma_{\mathrm{s}}^j} \quad . \tag{2.38}$$

Here, σ_i is the number of sites occupied by one particle of the species i and Γ_{s}^j is the surface site density of the specific surface type. For each subset $\mathbf{S}_{\mathrm{s}}^j$

$$\sum_{i \in \mathbf{S}_{\mathrm{s}}^j} \theta_i \quad = \quad 1 \tag{2.39}$$

must be fulfilled.

2.2 Chemical Reactions

2.2.1 General

General reaction equation

In general, a chemical reaction can be written as

$$\sum_{i \in \mathbf{S}} \nu_i' \mathrm{A}_i \quad \rightarrow \quad \sum_{i \in \mathbf{S}} \nu_i'' \mathrm{A}_i \quad , \tag{2.40}$$

where A_i is a symbol of the i-th species. ν_i' and ν_i'' are the stoichiometric coefficients of the reactants and products, respectively. There always exist integer coefficients. Their difference shall be denoted by

$$\nu_i \quad = \quad \nu_i' - \nu_i'' \quad . \tag{2.41}$$

For the set of all reactions we use the symbol $\mathbf{R}$.

Chemical equilibrium

For each reaction, there is always a reverse reaction, which leads to an equilibrium between the reactants and products:

$$\sum_{i \in \mathbf{S}} \nu_i' \mathrm{A}_i \quad \rightleftharpoons \quad \sum_{i \in \mathbf{S}} \nu_i'' \mathrm{A}_i \quad . \tag{2.42}$$

In case of equilibrium, the Gibbs Free Energy

$$G \quad = \quad H - TS \tag{2.43}$$

is a minimum for a given pressure and temperature. The specific Gibbs Energy can be defined using the specific enthalpy and entropy for a given pressure p^0 as

$$g_i = h_i - T s_i \quad . \tag{2.44}$$

The change of Gibbs Free Energy in a reaction that converts ν_i mol of each species is

$$\Delta_R \overline{G}^0 = \sum_{i \in \mathbf{S}} \nu_i g_i \,\mathrm{mol} \quad , \tag{2.45}$$

This quantity is related to the equilibrium constant

$$K_p = \prod_{i \in \mathbf{S}} \left(\frac{p_i}{p^0} \right)^{\nu_i} \tag{2.46}$$

by

$$K_p = \exp\left(-\frac{\Delta_R \overline{G}^0}{RT} \right) \quad . \tag{2.47}$$

In analogy, a equilibrium constant with respect to concentrations can be defined as

$$K_c = \prod_{i \in \mathbf{S}} c_i^{\nu_i} \quad . \tag{2.48}$$

Using the reference concentration c^0 at pressure p^0 we can write

$$K_c = \exp\left(-\frac{\Delta_R \overline{G}^0}{RT} \right) {c^0}^{\Delta_R \nu} \tag{2.49}$$

with

$$\Delta_R \nu = \sum_{i \in \mathbf{S}} \nu_i \quad . \tag{2.50}$$

For a gas mixture the ideal gas law yields

$$c^0 = \frac{p^0}{RT} \quad . \tag{2.51}$$

Reaction rate

The reaction rate in general depends on the concentrations of the reactants (probability of a collision) and on temperature (collision energy). Therefore, the dynamics of reaction can be written as:

$$\frac{\mathrm{d}c_i}{\mathrm{d}t} = \nu_i k_f \prod_{j \in \mathbf{S}} c_j^{\widetilde{\nu_j}'} \quad , \tag{2.52}$$

where k_f is a temperature-dependent rate coefficient (of the forward reaction) and $\widetilde{\nu_j}'$ are the reaction orders of each species. In analogy, we can write for the backward reaction:

$$\frac{\mathrm{d}c_i}{\mathrm{d}t} = -\nu_i k_r \prod_{j \in \mathbf{S}} c_j^{\widetilde{\nu_j}''} \quad . \tag{2.53}$$

The probability of a reaction due to a collision depends on the kinetic energy of the colliding particles. Therefore, *Arrhenius* chose the following ansatz for the rate coefficient:

$$k = A \cdot \exp\left(-\frac{E_a}{RT} \right) \quad . \tag{2.54}$$

The constant A is called pre exponential factor. However, sometimes there is an additional temperature dependence in this factor. Thus, we use the following expression:

$$k = A\, T^{\beta} \cdot \exp\left(-\frac{E_a}{RT} \right) \tag{2.55}$$

with an additional fit parameter β.

Elementary reactions

An elementary reaction is a reaction that can be observed on a molecular scale between single particles. Therefore, every complex chemical mechanism can (at least in theory) be broken down into elementary reactions. The advantage of such a scheme is that the coefficients in the Arrhenius equation get a physical meaning, which can be measured. The reaction orders equal the stoichiometric coefficients:

$$\widetilde{\nu}_j\prime = \nu_j\prime \qquad \text{bzw.} \qquad \widetilde{\nu}_j\prime\prime = \nu_j\prime\prime \quad . \tag{2.56}$$

In practical matters, collisions between three particles a relevant only.

The reaction is in equilibrium when the forward reaction (eq. 2.52) is as fast as the reverse reaction (eq. 2.53). Rearranging the terms, we get:

$$\frac{k_\mathrm{f}}{k_\mathrm{r}} = \prod_{j\in\mathbf{S}} c_j^{\nu_j} \quad . \tag{2.57}$$

Comparing with eq. 2.48 yields

$$K_\mathrm{c} = \frac{k_\mathrm{f}}{k_\mathrm{r}} \quad . \tag{2.58}$$

2.2.2 Gas-phase reactions

The set of all gas-phase reactions in an mechanism shall be denoted by $\mathbf{R}_\mathrm{g}$. These reactions only involve gas-phase species in $\mathbf{S}_\mathrm{g}$. If only the collision efficiency of a species matters, we can formally take it as a representant of the set $\mathbf{S}_\mathrm{M}$. In this case ν_i must be zero ($i \in \mathbf{S}_\mathrm{M}$).

The reaction rate of each gas-phase species is then computed as a sum over the reaction rates of each reaction:

$$\dot{\omega}_i = \frac{\mathrm{d}c_i}{\mathrm{d}t} = \sum_{k\in\mathbf{R}_\mathrm{g}} \nu_{ik}\, k_k \prod_{j\in\mathbf{S}_\mathrm{g}\cup\mathbf{S}_\mathrm{M}} c_j^{\widetilde{\nu}_{jk}\prime} \quad . \tag{2.59}$$

2.2.3 Pressure-dependent gas-phase reactions

Various dissociation and recombination reactions are treated like elementary reactions for simplicity. They always involve a third body in the collision process (symbol M). Due to complex interactions between the reactants, the reaction coefficient k depends on the concentration of M. According to the Lindemann theory, one can observe a direct proportionality in the low-pressure limit, where as in the high pressure limit we get saturation. Therefore we can define the two rate-limiting constants:

$$k_0 = \lim_{[\mathrm{M}]\to 0} \frac{k}{[M]} \quad . \tag{2.60}$$

and

$$k_\infty = \lim_{[\mathrm{M}]\to \infty} k \quad . \tag{2.61}$$

These two constants can be modeled by Arrhenius equations (eq. 2.55)

The Lindemann theory further yields for the rate coefficient with given concentrations:

$$k = k_\infty \left(\frac{p_\mathrm{r}}{1+p_\mathrm{r}}\right) F \quad , \tag{2.62}$$

where p_r is the reduced pressure

$$p_\mathrm{r} = \frac{k_0\,[\mathrm{M}]}{k_\infty} \tag{2.63}$$

and F the so-called *pressure fall-off blending function*. In the most simple case, the latter one equals unity. In DETCHEM, also the Troe formalism has been implemented to model this function:

$$\log_{10} F = \left[1 + \left(\frac{\log_{10} p_\mathrm{r} + c}{n - d\,(\log_{10} p_\mathrm{r} + c)}\right)^2\right]^{-1} \log_{10} F_\mathrm{cent} \quad . \tag{2.64}$$

With

$$c = -0,4 - 0,67 \log_{10} F_\mathrm{cent} \quad , \tag{2.65}$$

$$n = 0,75 - 1,27 \log_{10} F_\mathrm{cent} \quad , \tag{2.66}$$

$$d = 0,14 \tag{2.67}$$

and

$$F_\mathrm{cent} = (1-\alpha)\mathrm{e}^{-T/T^{***}} + \alpha\,\mathrm{e}^{-T/T^{*}} + \mathrm{e}^{-T^{**}/T} \quad . \tag{2.68}$$

The parameters α, T^{***}, T^{*} und T^{**} are the so-called Troe parameters and they are used to fit experimental data.

2.2.4 Surface Reactions

The description of surface reactions has been chosen in analogy to the gas-phase reactions. However, since there exist almost innumerable numbers of different surface structures, only little is know about the exact reaction paths. Instead of looking at interactions between single adsorbed species, we will use a mean-field approximation for the description of the surface. Thus, the adsorbate on the surface are described in terms of locally varying coverage fractions that represent concentrations of randomly distributed particles.

The set of surface reactions shall be denoted by $\mathbf{R}_\mathrm{s}$. In the case of the existence of more than on surface type, it can consist of several disjoint subsets $\mathbf{R}_\mathrm{s}^j \quad (j = 1, 2, \ldots)$.

The surface reaction rate is then given in analogy to eq. 2.59:

$$\dot{s}_i = \sum_{k \in \mathbf{R}_\mathrm{s}} \nu_{ik}\, k_k \prod_{j \in \mathbf{S}} c_j^{\widetilde{\nu}_{jk}\prime} \quad . \tag{2.69}$$

Using an Arrhenius expression (eq. 2.55), local interactions between adsorbate and their influence on the rate coefficients can be modeled in the mean-field approximation by an additional factor:

$$k_k = A_k\, T^{\beta_k} \cdot \exp\left(-\frac{E_{\mathrm{a},k}}{RT}\right) \cdot f_k\left(\{\theta_i\}\right) \quad . \tag{2.70}$$

Some adsorbate may change the energy potentials of the surface, which lead to a change of activation energies for some reactions. Furthermore, the probability of adsorption can be changed, which is associated with the pre-exponential factor. Therfore, the function f_k shall be chosen as follows:

$$f_k = \prod_{i \in \mathbf{S}_\mathrm{s}} \theta_i^{\mu_{ik}} \cdot \exp\left(\frac{\epsilon_{ik}\theta_i}{RT}\right) \quad . \tag{2.71}$$

Here μ_{ik} and ϵ_{ik} are additional model parameters.

A more convenient way to describe adsorption reactions is given in terms of sticking coefficients S_i^0. They basically quantify a probability ($0 \le S_i^0 \le 1$) that a particle hitting the surface is adsorbed. First of all, the probability depends on the existence of suitable adsorption sites. Furthermore, lateral interactions with other adsorbed species may have an influence. Therefore, one can define a local adsorption probability by

$$S_i^\mathrm{eff} = S_i^0 \cdot \prod_{j \in \mathbf{S}_\mathrm{s}} \theta_j^{\nu_{jk}\prime + \mu_{jk}} \quad . \tag{2.72}$$

The reaction rate $\dot{s}_i$ can be computed using the kinetic theory of gases by

$$\dot{s}_i = S_i^\mathrm{eff} \sqrt{\frac{RT}{2\pi M_i}}\, c_i \quad . \tag{2.73}$$

The latter equation assumes a Boltzmann distribution of molecular velocities near the surface. However, Motz and Wise showed that this assumption is not fulfilled for sticking probabilities close to unity. In this case the effiective sticking coefficient should be corrected by

$$S_i^\mathrm{MW} = \frac{S_i^\mathrm{eff}}{1 - \frac{1}{2} S_i^\mathrm{eff}} \quad . \tag{2.74}$$

2.3 Modeling of transport and surface reactions in washcoats

This section describes the washcoat models of DETCHEM to take into account the pore diffusion within a porous washcaot. DETCHEM provides two washcoat models. A simple model, which is based on the concept of effectiveness factors, and a detailed approach, which is based on solving reaction-diffusion equations within the washcoat.

2.3.1 Diffusion and reaction in porous catalysts

In a porous catalyst, the reaction occurs inside the pores: therefore the reactants must diffuse into the catalyst. As the reactants diffuse into the catalyst they encounter active sites and some of the reactants react. Therefore the reactant concentration declines as the distance into the catalyst increases, giving rise to a concentration gradient inside the catalyst. If the diffusion velocity is much lower than the intrinsic rate, the concentration gradients can be quite large. Also, the intrinsic rate depends on the gas phase concentration at the active sites. Hence, it is necessary to have a mean of calculation the effective rate of reaction at the gas-wall boundary, for a catalytic wall with variable concentration profiles inside.

2.3.2 Detailed washcoat model

The detailed washcoat model solves the reaction-diffusion equations in radial direction for every species i within the washcoat.

$$\frac{\partial j_i}{\partial r} - \gamma \dot{s}_i = 0 \tag{2.75}$$

The radial diffusion flux of species i is calculated by

$$j_i = -D_{\text{eff},i} \frac{\partial c_i}{\partial r} \tag{2.76}$$

where c_i = concentration of species i within the washcoat, γ = ratio of catalytic active area to washcoat volume and $D_{\text{eff},i}$= effective diffusion coefficient.

Implementation: In order to solve the reaction-diffusion equations, the washcoat is discretized according to the method of finite differences. The reaction rate at the gas-wall boundary is determined from the diffusion flux at the gas-wall boundary.

Application: The detailed washcoat model is a time consuming model. Depending on the reaction system, the calculation times can rise by one or two orders of magnitude. The benefit of this model is that it can be applied to every reaction system. Also, the concentrations and coverages within the washcoat can be calculated.

2.3.3 Effectiveness factor approach

Consider the reaction-diffusion equation (2.75) for a species with the folowing conditions:

- The species is consumed by a reaction with a first-order rate law, i.e. $\dot{s}_i = -kc_i$.
- The diffusion coefficient is constant.
- The washcoat is thick enough to assume zero concentration gradients at deepest point of the washcoat at position L.

In this case, there is an analytical solution of equation 2.75

$$c(r) = \frac{e^{-\phi}}{e^{\phi} + e^{-\phi}} c_{i,0} \cdot e^{\phi r/L} - \frac{e^{\phi}}{e^{\phi} + e^{-\phi}} c_{i,0} \cdot e^{-\phi r/L} \tag{2.77}$$

where the so-called Thiele modulus

$$\phi = L \sqrt{\frac{k}{D_{\text{eff},i}}} \tag{2.78}$$

has been used for convenience.

The diffusion flux at the boundary between the gas phase and the washcoat can be seen as an effective reaction rate $\bar{\dot{s}}_i$.

$$\bar{\dot{s}}_i = j_{i,0} = -\tanh(\phi) \cdot \sqrt{kD} c_{i,0} \tag{2.79}$$

This expression can be compared to the surface reaction rate $\dot{s}_i$ of the same amount of catalyst without diffusion limitations.

$$\dot{s}_i = -kLc_{i,0} \tag{2.80}$$

The ratio of the two rate expressions is called effectiveness factor η_i, which can be calculated analytically.

$$\eta_i = \frac{\bar{\dot{s}}_i}{\dot{s}_i} = \frac{\tanh(\phi)}{\phi} \tag{2.81}$$

Implementation: If the effectiveness factor model is used, first the reaction rate $\dot{s}_i$ is calculated. From this, the effective rate constant k, the Thiele modulus ϕ and the effectiveness factor η_i are calculated for a user-defined species. It is important that this species is representative for the consumption of all reactants. Choosing a species with vanishing concentration may yield non-realistic effectiveness factors. Since conservation of mass is required, all reaction rates of a surface are multiplied by the same effectiveness factor.

Application The effectiveness factor model is a very fast model. The calculation times are nearly the same with or without this model. Because of the application of the same effectiveness factor for the mean reaction rates of all species, the model can only be applied for reaction systems which are determined by the consumption of a major species.

2.3.4 Effective diffusion coefficients

In both washcoat models effective diffusion coefficients are needed. Depending on the pore size distribution of the washcoat different models can be used.

1. Knudsen diffusion coefficient

$$D_{\text{knud},i} = \frac{\epsilon_p}{\tau} \frac{d_p}{3} \sqrt{\frac{8RT}{\pi M_i}} \tag{2.82}$$

2. Molecular diffusion coefficient

$$D_{\text{eff},i} = \frac{\epsilon_p}{\tau} D_{\text{mol},i} \tag{2.83}$$

3 Mixed diffusion coefficient

$$D_{\text{eff},i} = \frac{\epsilon_p}{\tau} \bar{D}_i \tag{2.84}$$

$\bar{D}_i$ is defined as

$$\frac{1}{\bar{D}_i} = \frac{1}{D_{\text{mol},i}} + \frac{1}{D_{\text{knud},i}} \tag{2.85}$$

4. Random pore model

$$D_{\text{eff},i} = \bar{D}_{M,i} \epsilon_M^2 + \frac{\epsilon_\mu^2 (1 + 3\epsilon_M)}{1 - \epsilon_M} \bar{D}_{\mu,i} \tag{2.86}$$

where $\bar{D}_{M,i}$ and $\bar{D}_{\mu,i}$ is defined as

$$\frac{1}{\bar{D}_{M,i}} = \frac{1}{D_{\text{mol},i}} + \frac{1}{(D_{\text{knud},i})_M} \tag{2.87}$$

and

$$\frac{1}{\bar{D}_{\mu,i}} = \frac{1}{D_{\text{mol},i}} + \frac{1}{(D_{\text{knud},i})_\mu} \tag{2.88}$$

In all the above models ϵ_p, τ, d_p, M_i, $D_{\text{mol},i}$, $D_{\text{knud},i}$, $D_{\text{eff},i}$, ϵ_M, ϵ_μ represent the porosity, tortuosity, pore diameter, molecular weight of species i, Binary diffusion coefficient of species i, Knudsen diffusion coefficient of species i, Molecular diffusion coefficient of species i, porosity for macro pores and porosity for micro pores respectively.

Chapter 3

Input files

This chapter briefs about the input files used in DETCHEM models.
Other than the files described in this chapter there are model specific input files and the **CHEMINP** generated ***detchem.inp*** file. Model specific input files are explained in the specific chapters for each model and ***detchem.inp*** is explained in chapter 4.

3.1 General structure of inp-files

One of the major changes comparing version 2.0 with previous versions of DETCHEM is the syntax of input files. Except for the databases (***thermdata***, ***moldata*** and the mechanism files) all application specific data is now specified using the same format. These files should be assigned an application specific name with the extention ***.inp***.

The generic structure of such input files can be described as follows:

```
{command}
<tag1>
  option1 = value     # comment1
  option2   value     # comment2
  option3 = "string"  # comment3
  option4             # comment4
  <tag2>
    option5 = value
  </tag2>
</tag1>
```

There are several items:

- tags,
- options,
- values (numerical, strings or logical),
- commands, and
- comments.

All items must be separated by spaces, line breaks, commas (,), or equal signs (=). The number of delimiters are not significant. A line break is also required behind the last tag in any input file. Tags, options and commands do not distinguish small from capital letters, however in strings and names (e.g. species names) they are treated differently. The length of a line is limited to 999 characters, each option name can be up to 32 characters.

3.1.1 Tags

Tags are used to group data in the input file. They always consist of a pair in sharp brackets: `<tag> ... </tag>`. They define a hierarchy in the input file. Therefore do not forget the closing tag at the appropriate place. An error message will be generated in case of an unknown option or insufficient information given within the tag-environment.

3.1.2 Options and Values

The user defined data is specified using options. There are several types of options, most of them require a value, but some do not. Depending on the type of information, the values can be integers, floating point numbers, strings or logicals. Integers are numbers that may contain a negative sign, but no decimal point or exponent, whereas all FORTRAN type formats are accepted for floating point numbers (e.g. 0.0001, 1e-4, 1.d-04). Logical values are `yes` and `no` (or shorter `y` and `n`). All character combinations that do not contain any delimiting character can also be interpreted as strings. To avoid confusion or to include delimiting characters (especially blanks) or other special characters (#,!,=, etc.), use quotation marks.

In the later chapters we use a notation like in the following table to list the members of a tag.

<**TAG_NAME**>	tag
<*tag_name*>	tag with variable name
option1=*d*	floating point argument
option2=*i*	integer argument
option3=*b*	logical argument (yes or no)
option4=*s*	string argument
option5	option without argument
value	value without option name

3.1.3 Commands

The input file parser supports a few basic commands that help increasing the reusability of input files and debugging. Commands are always included in curly brackets as shown in the following example.

```
{verbose y}
{include filename1}
{include filename2}
{define temperature 523}
....
....
{ifdef temperature label1}
   T = 273               # use default temperature
{goto label2}
{label label1}
   T = {get temperature} # use defined temperature
{label label2}
....
....
```

`include` *filename* is used to include other files which are present in the working directory. This command can be used to increase readability and reusability of the input file. For instance, data commmon to several simulations like species input data can be saved in a separate file. By including the external file, it can be easily established that all simulations use the same data. An included file can also contain an include command. A maximum of five recursions is allowed.

`verbose` *yes_or_no* switch screen output on or off. When activated, the parser repeats the lines of the input files while the program reads them. This command is especially useful for debugging purposes. The user can find out which data was read and where the program has terminated in case an error occurred.

The parser also provides the possibility to define user variables in the input file. Variables can be decalred anywhere in the input file using the command **`define`** *variable_name value*. The *variable_name* can be any combination of letters (case sensitive), numbers and some special characters (e.g., _,+,−). *value* can also be any expression whout blank characters. If the parser later finds the commands **`get`** *variable_name*, it will be replaced by the content of *value*. Depending on the context where the variable is inserted, *value* can be interpreted as string or numeric value. Each variable can only be defined once. Attempting to redefine a variable will create an error message.

For numeric variables the **`get`** command can also perform simple arithmetic calculations. In this case, use it in the form

{ **`get`** *v1 operator v2* }

Here, *v1* and *v2* are either constant number expressions (e. g. 5, 1e10, -0.003) or variable names containing numerical values. The operators can be chosen from `+`, `-`, `*` and `/` . Please note to separate arguments and operator by at least one blank character.

A series of numeric values (e.g. for temperature inputs) can be generated using the **`do`** command. It can be used in the forms

{ **`do`** *value_from value_to* }

or

{ **`do`** *value_from value_to step_size* }

Here, *value_from*, *value_to*, and *step_size* are either constant number expressions or variable names containing numerical values. If *step_size* is omitted, a step size of 1 is used.

Example:

```
{define Tstep 25}        # define a variable
...
<TPROFILE>
  {do 300 400 Tstep}     # is replaced by: 300 325 350 375 400
</TPROFILE>
....
```

The pair commands **goto** *label_name* and **label** *label_name* allows skipping of sections in the input file. On encounter of a **goto** command all following lines will be ignored until the matching **label** command is found. That implies that only forward jumps are possible.

A simple branching is possible by the command **ifdef** *variable_name label_name*. This command checks if a variable *variable_name* is defined without looking at its value. If yes, **goto** *label_name* is executed. Otherwise the parser continues at the next available statement.

3.1.4 Comments

Text in one line preceeded by a # or ! is treated as comment statements in the input file. We recommend the use of # over the other one.

3.1.5 Units

In DETCHEM version 2.3 a new feature was added to DETCHEM input files. Most physical properties, which are usually treated to be given in SI-units, can automatically be converted by the input file parser from arbitrary units into the required standard unit. Where possible, a check for consistency of the unit is made. Thus, by stating a unit explicitly in the input file, you can avoid confusion of physical properties. For instance, a step size required in units of time (i. e. seconds) will be refused if the value is given in meters.

For instance, an input file could consist of the following lines:

```
<myproperties>
  p=1[bar]        # pressure
  V=5[L]          # volume
  T=473[K]        # temperature
  Vdot=2[L/min]   # volume flux
  Cp=1.5[kJ/kg/K] # heat capacity
</myproperties>
```

Thus, a unit follows a numerical value using square brackets [ ]. You can leave blanks between the numerical value and the brackets, but the unit must be on the same line of the input file.

The unit symbol is case sensitive. While **[mm]** stands for 10^{-3} meters, **[Mm]** means 10^6 meters. Each single unit symbol can be followed by an integer exponent, e. g. a cubic centimeter is written as **[cm3]**. To multiply two unit symbols, simply separate them by a blank character or use the symbol *****. That is, both **[N m]** and **[N*m]** symbolize the unit Newton-meter. The slash charcater **/** is used to indicate that the next unit appears in the denominator of a fraction. However, notice that it only applies to one following unit symbol. In order to specify the unit J/(kgK) write **[J/kg/K]** or **[J kg-1 K-1]**.

SI-units usually can be prefixed to abbreviate powers of ten. If a unit is prefixable, you can use the prefixes **h** for 10^2, **k** for 10^3, **M** for 10^6, ... (**G**,**T**,**P**,**E**,**Z**,**Y**); and **d** for 10^{-1}, **c** for 10^{-2}, **m** for 10^{-3}, **u** for 10^{-6}, ... (**n**,**p**,**f**,**a**,**z**,**y**). Note that micrometers are written as **[um]** instead of **[μm]**.

Usually DETCHEM expects temperatures in Kelvin **[K]**. Now you can also use Celsius **[°C]** or Fahrenheit **[°F]**. Note however that the parser does not check if the use of these temperature units is allowed in a specific context. When the program expects a temperature difference, where you enter **10[°C]**, the program will use a step size of 283.15 K! Also notice that the use of °C and °F is not recommended in compound units, e. g. always use **[J/K]** instead of **[J/°C]**.

The folowing table lists the known unit symbols:

Quantity	Unit	Prefixable
None	1	
	1 % = 10^{-2}	
	1 ppm = 10^{-6}	
	1 ppb = 10^{-9}	

Quantity	Unit	Prefixable
Length	1 m	yes
	1 Angstrom = 10^{-10} m	
	1 in = 25.4 mm	
	1 mil = 1/1000 in	
	1 ft = 12 in	
	1 yd = 3 ft	
	1 mile = 1760 yd	
Volume	1 l = 1 L = 1 dm^3	yes
	1 imp.gal = 277.42 in^3	
	1 imp.qt = 1/4 imp.gal	
	1 imp.pt = 1/2 imp.qt	
	1 imp.floz = 1/20 imp.pt	
	1 USgal = 231 in^3	
	1 USqt = 1/4 USgal	
	1 USpt = 1/2 USqt	
	1 USfloz = 1/16 USpt	
Time	1 s	yes
	1 min = 60 s	
	1 h = 60 min	
	1 d = 24 h	
Mass	1 kg	yes
	1 lb = 453.59237 g	
	1 oz = 1/16 lb	
Temperature	1 K	
	0 °C = 273.15 K	
	32 °F = 0 °C ; 212 °F = 100 °C	
Quantity	1 mol	yes
Frequency	1 Hz = 1 s^{-1}	yes
Force	1 N = 1 kg m s^{-2}	yes
	1 lbf = 4.44822 N	
Pressure	1 Pa = 1 N/m^2	yes
	1 bar = 10^5 Pa	yes
	1 mmHg = 101325/760 Pa	
	1 psi = 1 lbf/in^2	
Energy	1 J = 1 N m	yes
	1 cal = 4.1868 J	yes
Power	1 W = 1 J/s	yes
Electrical Current	1 A	yes
Voltage	1 V = 1 W/A	yes
Electrical Resistance	1 Ohm = 1 V/A	yes
Electrical Charge	1 C = 1 A s	yes

3.2 thermdata

The thermodynamic database contains enthalpy, entropy and heat capacity data for each species in the NASA Equilibrium code [10]. The calculation of thermodynamic data from the coefficients $a_1 \dots a_7$ is described by equation 2.13, 2.16 and 2.17. There are two temperature intervals with different sets of coefficients to achieve better accuracy. The first seven entries are the values $a_1 \dots a_7$ for the temperature interval $T_{\text{jump}} < T < T_{\text{high}}$, the next seven entries are the values $a_1 \dots a_7$ for the temperature interval $T_{\text{low}} < T < T_{\text{jump}}$.

```
----------------------------------------
Example 3.1.1: thermdata (selection)
----------------------------------------------------------------
H2               J 3/77 H   2    0    0    0G   300.000  5000.000
 0.30667095E+01 0.57473755E-03 0.13938319E-07-0.25483518E-10 0.29098574E-14
-0.86547412E+03-0.17798424E+01 0.33553514E+01 0.50136144E-03-0.23006908E-06
-0.47905324E-09 0.48522585E-12-0.10191626E+04-0.35477228E+01
----------------------------------------------------------------
```

The following format is used for the section above, selected from the ***thermdata*** file.

line 1: The first 8 characters in this line are used to define the chemical species with the format specification(A8),which is followed by comments with (A16) format. The next 20 columns are reserved to specify the atoms present in the defined chemical species with the format 4(A5). In the above example H represents the atom and 2 stands for the number of atoms present. The next 3 columns are reserved for the species type, but not evaluated in the present version. The rest of the columns are used to define the temperature intervals as follows. Low temperature T_{low} (E10.0) and high temperature T_{high} (E10.0), jump temperature T_{jump} (1000K unless specified)(E8.0).

line 2-4: Coefficients $a_1 \ldots a_7$ for upper (first seven rules) and lower (next seven rules) temperature interval (5(E15.8) per line).

In the case of reversible reactions this data is used to calculate the kinetic data for the reverse reaction. And also the thermodynamic data for the gas-phase species are used to calculate the reaction enthalpies. If the thermodynamic data of the adsorbed species are unknown and if these species are not involved in reversible reactions, dummy values (e.g 0.0) can be given. A section of ***thermdata*** file with dummy values is shown below.

```
-----------------------------------------------
 Example 3.1.2: thermdata (selection)
-------------------------------------------------------------------------------
H2O(PD)          92491  O   1H   2PD  1     I    300.00   3000.00 1000.00
0.00000000E+00 0.00000000E+00 0.00000000E+00 0.00000000E+00 0.00000000E+00
0.00000000E+00 0.00000000E+00 0.00000000E+00 0.00000000E+00 0.00000000E+00
0.00000000E+00 0.00000000E+00 0.00000000E+00 0.00000000E+00
-------------------------------------------------------------------------------
```

3.3 moldata

The ***moldata*** file contains the molar mass and kinetic theory parameters of chemical species.

```
-----------------------------------------------
Example 3.2.1: moldata (selection)
-------------------------------------------------------------------------
H2    1   2.016    38.000    2.920   0.000   0.790   280.000 Wa.(82)
-------------------------------------------------------------------------
```

A section of ***moldata*** is shown above, where the entries has the following meaning

H2	Species Symbol
1	Index indicating whether the molecule has a monoatomic, linear or non-linear geometrical configuration (0= single atom, 1= linear, 2=nonlinear)
2.016	molar mass in g/mol
38.000	Lennard-Jones potential well depth ε/k_{B} in K
2.920	Lennard-Jones collision diameter σ in Angstroms
0.000	Dipole moment μ in Debye
0.790	Polarizability α in cubic Angstrom
280.000	Rotational relaxation collision number Z_{rot} at 298 K
Wa.(82)	Comment

Each species occurring the mechanism files (described later) must be contained in this database, otherwise the program will formally fail. Except the molar mass data all other data do not have any meaning for surface species.

3.4 Gas-phase chemistry mechanism file

The gas-phase chemistry mechanism file lists the elementary reactions occurring in the gas-phase.This file can be give any name, and the same should be mentioned in the ***mech.inp*** file (description follows). For convenience, the user is advised to copy and modify the lines, while keeping the same format. An example of gas-phase reaction mechanism file is shown below.

```
------------------------------------
Example 3.3.1: noo2m
------------------------------------------------------
MECHANISM OF THE NO-O2 REACTION (P. KLAUS 1997)
 ****
O      +O     +M(1)   =O2    +M(1)    2.900E+17    -1.000   0.0
NO2    +M(1)  =NO     +O     +M(1)    1.100E+16     0.0   275.88
NO2    +O     =NO     +O2             1.000E+13     0.0     2.51
NO2    +NO2   =NO     +NO    +O2      1.600E+12     0.00  109.19
END
000 COMPLEX REACTIONS
END
COLLISION EFFICIENCIES:
M(1)    =O2    +N2
         0.40   0.40
END
------------------------------------------------------
```

In the above example each reaction is given in one line. Irreversible reactions (denoted by >) as well as reversible ones (denoted by =) can be specified. The reaction equations can have a maximum of 3 species on the left and three species on the right side of the equation, however the total number of species in an equation is not allowed to exceed five. A species can contain a maximum of 8 characters at maximum followed by a separator (+,>,=) unless the end of the equation is reached. Then the kinetics data follows. The format of the whole line is: 5(A8,A1), E10.3, F7.1, F10.1. Unless familiar with FORTRAN nomenclature, just copy and modify wherever appropriate, keeping the given format.

The reaction rate is calculated by using concentrations in the units mol/cm^3. The rate coefficients are given in terms of modified Arrhenius expression as stated in equation 2.55. The input order is assigned as follows.

A_k value in the first column in cm, s, mol (according to the reaction order.
β_k value in second column.
E_a value in third column in kJ/mol For reversible reactions, the reverse reaction rates are derived from the equilibrium constants using Equations 2.48. All of the reaction orders are according to the stoichiometry of the reaction.

The symbol M(1) in the mechanism denotes third bodies as described in section 2.1.3. The last part of the *gas-phase mechanism* file denotes the collision efficiencies for these third bodies, if reaction with third bodies are specified. The collision are by default unity for the species not given in the list. In example 3.3.1, the collision efficiencies of the species O_2 and N_2 are modified to be 0.4 instead of unity. Information on proper collision efficiencies will be provided in connection with reaction mechanisms. If the collision efficiencies for all species in a reaction are unity, them plain symbol M can be used.

```
------------------------------------
Example 3.3.2:
---------------------------------------------------------------------
H     +CH2O          +M(2) =CH2OH    +M(2)   0.540E+12   0.5   15.070
                                     LOW    1.270E+32 -4.820 27.32
                                     TROE .7187 103.0 1291.0 4160.0
---------------------------------------------------------------------
```

Aside from the conventional way to model the reaction rate by an *Arrhenius* expression, a *Troe* expression can be used, which is very useful to include the additional pressure dependence of recombination and dissociation reactions. The rate law here becomes more complicated; the 3.3.2 illustrates the format used to define *Troe reactions* by ten parameters. The data in the first line are A_∞, β_∞, $E_{a\infty}$, the data in the line behind the LOW keywords are A_0, β_0, E_{a0} and the data behind the TROE keyword are a, T^{***}, T^*, T^{**}. If a = 0.5 and T^{***}= T^* = 1E30 then a Lindemann expression is used where the broadening factor F is unity.

3.5 Surface chemistry mechanism file

This file lists the surface elementary reaction mechanism. This file can be given any name and the same must be mentioned in the ***mech.inp*** file (described later). An example of the surface mechanism file is shown below. For convenience, the user is advised to copy and modify the lines, while keeping the same format. An example of surface chemistry mechanism file is shown below.

```
------------------------------
Example 3.4.1: noo2sm
----------------------------------------------------------------------
SURFACE MECHANISM OF THE NO-O2 REACTION
****
**********************************************************************
****                                                                 *
**** NO-O2 SIMPLE TEST SURFACE MECHANISM ON PT                       *
****                                                                 *
**********************************************************************
**** 1. ADSORPTION
STICK
O2       +PT(s)       +PT(s)     >O(s)    +O(s)    7.000E-02  0.0    0.0
STICK
NO       +PT(s)       >NO(s)                       8.500E-01  0.0    0.0
STICK
NO2      +PT(s)       >NO2(s)                      9.000E-01  0.0    0.0
STICK
O        +PT(s)       >O(s)                        1.000E-00  0.0    0.0
***** 2 DESORPTION
O(s)     +O(s)        >PT(s)     +PT(s)   +O2      3.700E+21  0.0  213.0
$O(s)                                             0.0        0.0   70.0
NO(s)    >NO          +PT(s)                       1.000E+16  0.0   90.0
NO2(s)   >NO2         +PT(s)                       1.000E+13  0.0   60.0
***** 3 SURFACE REACTION
NO(s)    +O(s)        >NO2(s)   +PT(s)             3.700E+21  0.0   96.3
$O(s)                                             0.0        0.0   70.0
$NO(s)                                            0.0        1.0   70.0
NO2(s)   +PT(s)       >NO(s)    +O(s)              3.700E+21  0.0   79.57
END
----------------------------------------------------------------------
```

The first line of the mechanism must begin with SURF.

The mechanism is given in terms of elementary reactions. Each reaction is given in one line with possible extension lines, one above and/or one below this line. Irreversible reactions (denoted by >) as well as reversible ones (denoted by =) can be specified. The reaction equations can have a maximum of three species on the left and three species on the right side of the equation, however the total number of species in one equation is not allowed to exceed five. A species can contain a maximum of 8 characters followed by a separator (+, >, =) unless the end of the equation is reached. Then, the kinetic data follow. The format of the whole line is: 5(A8,A1), E10.3, F7.1, F10.1. Unless familiar with FORTRAN nomenclature, just copy and modify wherever appropriate, keeping the given format.

The reaction rate is calculated by using concentrations with the units mol/cm^3 in the gas phase and mol/cm2 on the surface. The rate coefficients are given in terms of a modified Arrhenius expression as stated in Equation 2.70. The input order is assigned as follows:

A_k value in the first column (cm,s or mol according to order of the reaction).
β_k value in the second column.
E_a value in the third column kJ/mol.

For reversible reactions, the reverse reaction rates are derived from the equilibrium constants using Equation 2.48

STICK means that a sticking coefficient is given (first column in the subsequent line) for the reaction in the next line. The given coefficient is the initial sticking coefficient S_0. It can be modified by additional parameters β and E_a.

By default, all of the reaction orders are according to the stoichiometry of the reaction, for instance the reaction

```
STICK
O2       +PT(s)       +PT(s)    >O(s)    +O(s)   7.000E-02   0.0   0.0
```

is second order in vacancies (PT(s)) and first order in O2. Here, the value 7.000E-02 is the initial sticking coefficient (= 0.07).

The order of the reaction can be modified, and the activation energy can be made coverage dependent by using the parameters μ_{ik} and ϵ_{ik} in Equation 2.70. The modified data can be given in a line below the reaction equation; this line must start with a symbol followed by the species that the modification refers to (species i in

Equation 2.70). Several modifications can be used as shown in Example 3.4.1. The value in the second column modifies the order of the reaction (μ_{ik}), the change to the stoichiometric value is specified here. The value in the third column (ϵ_{ik}) modifies the activation energy. In the following reaction,

```
STICK
H2     +PT(s)    +PT(s)    >H(s)    +H(s)    0.046E-00   0.0   0.0
$PT(S)                                                  -1.0   0.0
```

STICK in the line above the reaction means that 0.046 is the initial sticking coefficient, PT(s) below the reaction line means that the rate coefficients/order of the reaction have an additional coverage dependence in the following way: PT(s) gives the related species (here: PT(s) = vacancies), and a nonzero value in second column (-1.0) modifies the order by -1 for this species, i.e. the reaction order in PT(s) is now first order and not second order. In the following reaction,

```
O(s)   +O(s)     >PT(s)     +PT(s)  +O2      3.700E+21   0.0 213.0
$O(s)                                                    0.0  70.0
```

A is 3.7x10^{21} cm^2/(mol*s), β is 0, and E$_a$ is 213 kJ/mol. O(s) below the reaction line means the rate coefficients/order of the reaction have an additional coverage dependence in the following way (Equation 2.70): The nonzero value in the third column (70.0) reduces the activation energy by 70 kJ/mol if the surface is completely covered with the given species, here O(s).

3.6 Species input

For any DETCHEM application the user needs to define the species that are to be considered. In many cases it is advisible to put the declaration in a separate file, e. g. ***species.inp***, and bind it to the application's input file by an an include statement

```
{include species.inp}
```

The **<SPECIES>** section contains gas-phase species as well as surface species. Always there must be exactly one gas-phase ensemble, where as you can have any number of surface ensembles (for reasons of memory allocation the number is usually limited to 5). In addition, third-body species can be defined in this section. An example is shown below.

```
<SPECIES>
  <GASPHASE>
    NO    O2    NO2    O    N2
  </GASPHASE>

  <SURFACE name="Pt" mol/cm2 = 2.72e-9>
    PT(s)    O(s)    NO2(s)    NO(s)

    <INITIAL>
      PT(s)    *
      O(s)     0.01
      NO(s)    0.08
    </INITIAL>
  </SURFACE>

  <THIRD-BODY>
    M = 0.4 O2 + 0.4 N2
  </THIRD-BODY>
</SPECIES>
```

In order to distinguish different surface types (e.g. Platinum, Rhodium etc.) you can assign an optional string parameter **name** inside the **<SURFACE>** tag (up to 8 characters). Further, for each surface a surface site density must be defined. It can be given in moles per square centimeter or moles per square meter. The initial coverage of surface species must be specified under element tag **<INITIAL>**. While specifying the initial coverage, one species can be given * value. In this case the value assigned to the particular species would be 1-(sum of initial coverage of rest of the species). In the example given above Pt(s) would be assigned a value of 0.91. Surface species having zero initial coverage need not be specified.

The tag **<THIRD-BODY>**, which is optional can be used to define reactions involving third bodies. This option is especially useful while defining global reactions with third bodies. In the above example M stands for the

third body (It can be any alphabet) and 0.4 is the collision efficiency. If the third body is already defined in the mechanism file, then the definitions of the mechanism will be taken. In short the third body definitions in the mechanism file over-rides the definitions in the **<SPECIES>** section.

Species occupying more than one site

There are instances where an adsorbed (surface) species can occupy more than one site on the given surface. For example some hydrocarbon species have size which needs more than one site to be adsorbed on the surface. In such cases, the information should be specified in the **<SURFACE>** section as shown below.

```
<SURFACE mol/cm2 = 2.7E-09>
  PT(s)
  O(s)
  NO2(s) * 2
  NO(s)
  <INITIAL>
    PT(s)    *
    O(s)     0.01
    NO(s)    0.08
  </INITIAL>
</SURFACE>
```

In the above example NO2(s) occupies 2 surface sites.

Options in <SPECIES> section

<**GASPHASE**>	gas-phase species list
<**SURFACE**>	surface species list
<**THIRD-BODY**>	third-body species list

Options in <SURFACE> subsection

name=*s*	surface name, e.g. 'Pt', 'Rh', etc.
mol/cm2=*d*	surface site density [mol/cm^2]
mol/m2=*d*	surface site density [mol/m^2]
***=*i*	number of surface sites occupied by one molecule of species (default: 1)
species name	species name
<**INITIAL**>	initial coverages

3.7 Mechanism input

Most DETCHEM applications deal with chemical reaction mechanisms. Therefore the reaction system needs to be declared in the application input file. This is done in the **<MECHANISM>** section of the input file. An example using the classic style of mechanism input files is shown below:

```
<MECHANISM>
  <GASPHASE>
    file="gasphase"
  </GASPHASE>

  <SURFACE name="Pt">
    file="surface"
  </SURFACE>
</MECHANISM>
```

In this case the **<MECHANISM>** tag only acts as a name holder for the external mechanism files. Due to compatibility with older DETCHEM versions, such mechanism files do not follow the new input format and therefore they can not be included directly into the input file. These files must be refered to by using the **file** option.

If several apllications shall use the same sets of reactions, it is advisible to put the mechanism declaration into a separate file, e. g. ***mech.inp***, and inlcude it in the application input file by

```
{include mech.inp}
```

DETCHEM can be used with zero or one gas-phase mechanism and any number (memory allocation defaults to five) of surface mechanisms. In order to distinguish different surface mechanisms, the **name** member can be assigned a string of up to 8 characters. A mechanism is enclosed within a pair of **<GASPHASE>** or **<SURFACE>** tags. However, each mechanism can can constist of an arbitrary number of mechanism files, e. g.

```
<MECHANISM>
  <GASPHASE>
    file="gasphase"
  </GASPHASE>

  <SURFACE name="Pt">
    file="surface1"
    file="surface2"
  </SURFACE>

  <SURFACE name="Rh">
    file="surface3"
  </SURFACE>

  <SURFACE name="Ni">
    file="surface4"
  </SURFACE>
</MECHANISM>
```

Note: Now, the use of external mechanism files is not mandatory anymore. Each reaction can also be defined directly in the input file using the format free input file syntax.

Gasphase mechanism

The **<GASPHASE>** tag can hold the following types of input:

file=*s*	gas-phase mechanism file
<**REACTION**>	elementary-step reaction
<**TROE**>	Troe reeaction
<**GLOBAL**>	global reaction

An external gas-phase mechanism file as descibed in chapter 3.4 is specified by the **file** option. Alternatively or in addition, the user can define elementary-step reactions with Arrhinus or Troe rate laws or complex (global) reactions with arbitrary rate laws. The syntax will described later in this chapter.

Surface mechanisms

The **<SURFACE>** subsection may contain the following options:

name=*s*	surface mechanism name
file=*s*	surface mechanism file name
<**REACTION**>	elementary-step reaction
<**STICK**>	stick reaction
<**GLOBAL**>	global surface reaction

One or more external surface mechanism files (see chapter 3.5) can specified by the **file** option. Alternatively or in addition, the user can define stick reactions, elementary-step reactions with Arrhinus rate laws or complex (global) reactions with arbitrary rate laws. The syntax will described later in this chapter.

If the logical option **MotzWise** is set, then the Motz-Wise correction for stick reactions will be applied.

<REACTION> – elementary step reactions

Both, gas-phase and surface reactions can be declared in the input file by using the **<REACTION>** tag. Each tag holds one reaction and its reaction parameters as shown in the example below:

```
<REACTION>
   NO2 + NO2 = NO + NO + O2
   A/cm_units = 1.6e12
   Ea/kJ_mol  = 109.19
</REACTION>
```

First, the reaction equation is written in a natural way. It is only required to separate species names and operators by at least one blank character. Each reaction can contain up to 3 reactants and up to 3 products. (The limitation to a total of 5 species of the formatted input files does not apply.) However, stoichiometric coefficients cannot be applied (see also input for global reactions further below). Use a + sign to separate two reactants or two products. The > and = symbols are used for a non-reversible and reversible reactions respectively.

Then, the rate coefficients need to be declared. We assume an Arrhenius type of rate law with a pre-exponential factor A, a temperature exponent β and an activation energy E_a. The pre-exponential factor and the activation energy can be given in a variety of units. The possible options are summarized below:

A/SIunits=*d*	pre-exponential factor [mol, m, s, K]
A/cm_units=*d*	pre-exponential factor [mol, cm, s, K]
A/molefracs=*d*	pre-expontential factor w.r.t. mole fractions [mol/m3 s, K]
beta=*d*	temperature exponent (default: 0)
Ea/J_mol=*d*	activation energy [J/mol]
Ea/kJ_mol=*d*	activation energy [kJ/mol]
Ea/R=*d*	activation energy / R [K]
cflag=*s*	character flag
<**COV**>	coverage dependency

The rate expression for surface reactions can further be modified by coverage dependent factors. Such additional factors are defined using the **<COV>** tag, the details of which are given later in this chapter.

The character flag is only used for special applications like sensitivity analysis. It will be described in detail where necessary.

<TROE> – pressure dependent reactions

Pressure dependent gas-phase reactions may use Troe parameters. The declaration is similar to reactions with Arrhenis parameters.

```
<TROE>
   H2O2 + M1 = OH + OH + M1
   A_inf/cm_units = 4e11
   Ea_inf/kJ_mol  = 155
   A_0/cm_units   = 4e16
   Ea_0/kJ_mol    = 183
   alpha          = 0.62
   T***           = 58
   T*             = 2700
   T**            = 21700
</TROE>
```

The reaction equation is written in the same way as explained before. A third-body species must appear in the reaction equation on both sides. Then, the two rates coefficients k_∞ and k_0 are defined by their corresponding Arrhenius parameters. Finally, the four Troe parameters need to be defined. The available parameters are summarized below:

A_inf/SIunits=*d*	pre-exponential factor [mol, m, s, K]
A_inf/cm_units=*d*	pre-exponential factor [mol, cm, s, K]
beta_inf=*d*	temperature exponent (default: 0)
Ea_inf/J_mol=*d*	activation energy [J/mol]
Ea_inf/kJ_mol=*d*	activation energy [kJ/mol]
Ea_inf/R=*d*	activation energy / R [K]
A_0/SIunits=*d*	pre-exponential factor [mol, m, s, K]
A_0/cm_units=*d*	pre-exponential factor [mol, cm, s, K]
beta_0=*d*	temperature exponent (default: 0)
Ea_0/J_mol=*d*	activation energy [J/mol]
Ea_0/kJ_mol=*d*	activation energy [kJ/mol]
Ea_0/R=*d*	activation energy / R [K]
alpha=*d*	Troe parameter alpha
T*=*d*	Troe parameter T*** [K]
T*=*d*	Troe parameter T* [K]
T**=*d*	Troe parameter T** [K]
cflag=*s*	character flag

The character flag is only used for special applications like sensitivity analysis. It will be described in detail where necessary.

<STICK> – surface adsorption reactions

Adsorption reactions are often described in terms of a sticking probability. Such reactions are declared as in the following example:

```
<STICK>
   CO + (s) = CO(s)
   S0 = 0.9
</STICK>
```

The reaction equation is written in a natural way. It is only required to separate species names and operators by at least one blank character. Each reaction can contain up to 3 reactants and up to 3 products. However, stoichiometric coefficients cannot be applied (see also input for global reactions further below). Use a + sign to separate two reactants or two products. The > and = symbols are used for a non-reversible and reversible reactions respectively. A stick reaction must contain exactly one gas-phase species on the left-hand side of the reaction equation.

As a reaction parameter, only the sticking coefficient S_0 is required. Additional parameters are summarized below:

S0=*d*	sticking coefficient
beta=*d*	temperature exponent (default: 0)
Ea/J_mol=*d*	activation energy [J/mol]
Ea/kJ_mol=*d*	activation energy [kJ/mol]
Ea/R=*d*	activation energy / R [K]
cflag=*s*	character flag
<**COV**>	coverage dependency

The rate expression can further be modified by coverage dependent factors. Such additional factors are defined using the **<COV>** tag, the details of which are given later in this chapter.

The character flag is only used for special applications like sensitivity analysis. It will be described in detail where necessary.

<COV> – coverage dependend factors

The rate expression of surface reactions can be modified by coverage dependent factors of the form

$$f(\theta_i) = \theta_i^{\mu_i} \exp\left(\frac{\epsilon_i}{RT}\right) \tag{3.1}$$

The parameters μ_i and ϵ_i for a surface species i can be defined using the **<COV>** tag within a definition of a surface reaction.

```
<REACTION>
   O(s) + O(s) > (s) + (s) + O2
   A/cm_units = 3.7e21
   Ea/kJ_mol  = 213
   <COV> O(s) epsilon/kJ_mol=70 </COV>
</REACTION>
```

The **<COV>** tag first lists the name of the surface species that appears as the argument of the modifying function. Then the parameters μ_i and ϵ_i can be defined. If one of them is omitted, its value defaults to zero. For the change of activation energy varaious units may be used. All possible options are summarized in the table below.

(species name)	name of surface species
epsilon/J_mol=*d*	change of activation energy [J/mol]
epsilon/kJ_mol=*d*	change of activation energy [kJ/mol]
epsilon/R=*d*	change of activation energy / R [K]
mu=*d*	change of reaction order

<GLOBAL> – complex (global) reactions

Sometimes, more complex reactions that are no elementary reactions need to be defined. Due to the strict format limitations of the gas-phase and surface mechanism files, they cannot be declared using the external mechanism files. Global reactions were introduced in DETCHEM version 2.0 and are to be declared in the the input file directly. An example of a global gas-phase reaction is shown below:

```
<MECHANISM>
  <GASPHASE>
    <GLOBAL>

      2 CO + O2 > 2 CO2

      <ARRHENIUS>
```

```
        CO  1.1
        A/cm_units = 1E10
        beta = 0
        Ea/kJ_mol = 275
      </ARRHENIUS>

    </GLOBAL>
  </GASPHASE>
</MECHANISM>
```

This input defines a reaction

$$2\,\mathrm{CO} + \mathrm{O_2} \longrightarrow 2\,\mathrm{CO_2} \tag{3.2}$$

with the rate law

$$\frac{\mathrm{d[O_2]}}{\mathrm{d}t} = -10^{10}\mathrm{cm^3/mol\,s}\,\exp\left(-\frac{275\mathrm{kJ/mol}}{RT}\right)[\mathrm{CO}]^{1.1}\,[\mathrm{O_2}] \tag{3.3}$$

In the global section, the user can specify any chemical reaction. Just write the reaction equation in a natural way:

component1 + component2 + ...> component3 + component4 + ...

for forward reactions or

component1 + component2 + ...= component3 + component4 + ...

for reversible reactions. Each component consits of an optional stoichiometric coefficient and a species name. The stoichiometric coefficient must be integer, if omitted it defaults to 1. Stoichiometric coefficients, species names and **+** signs must be separated by space characters.

The rate law can be defined in two ways. First, the user can specify the coefficients of an Arrhenius rate law. The possible parameters in the **<ARRHENIUS>** subsection are

A/SIunits=*d*	pre-exponential factor [mol, m, s, K]
A/cm_units=*d*	pre-exponential factor [mol, cm, s, K]
A/molefracs=*d*	pre-expontential factor w.r.t. mole fractions [mol/m3 s, K]
beta=*d*	temperature exponent (default: 0)
Ea/J_mol=*d*	activation energy [J/mol]
Ea/kJ_mol=*d*	activation energy [kJ/mol]
Ea/R=*d*	activation energy / R [K]
species symbol=*d*	reaction order of species

So, the user can specify the preexponential factor A, the temperature exponent β, and/or the activation energy E_A. The reaction order of a species defaults to the reactant's stoichiometric coefficient. If stoichiometric coefficient and reaction order of a species shall be different, then specify the species name and its reaction order within the **<ARRHENIUS>** subsection.

Another way to define a rate law of a global reaction is a user defined rate law. This can be declared in the following way:

```
<MECHANISM>
  <GASPHASE>
    <GLOBAL>

      2 NO + O2 > 2 NO2
      userID = 1

    </GLOBAL>
  </GASPHASE>
</MECHANISM>
```

Using **userID**, an integer number is assinged to the global reaction, which identifies the corresponding rate in the source code. Then the user needs to provide FORTRAN code for the rate law. The interface routines are contained in the DETCHEM distribution in file ***lib_detchem/DCRGU_UserSub.f*** for gas-phase mechanisms and ***lib_detchem/DCRSU_UserSub.f*** for surface mechanisms. Before running the application, the code must be recompiled with your user program code.

An exaple for user-defined gasphase rate-law is shown below:

```
c     ************************************************************

      subroutine DCRGU_Init
c     this subroutine is called once upon initialization of the
c     DETCHEM-Library
c     use this subroutine to make your own initialization of
c     variables in the common block declaration
c     e.g., it is advisable to store species indices by calling
c     the supplied functions of the DETCHEM-Library
```

```
      include 'DCRGU_UserData.F'

      call DCS_GetIndex('O2',o2)
c      if (o2 .le. 0) then ...(error)
      call DCS_GetIndex('N2',n2)
      call DCS_GetIndex('NO',no)
      call DCS_GetIndex('NO2',no2)

c     note:
c     In DETCHEM reactions module species are always identified by
c     their index in the list of all species (DCS)
c     In the rare case that you want the index according to the
c     gas-phase (DCSG) or surface (DCSS) species list,
c     you must call DCSG_GetIndex oder DCSS_GetIndex, respectively.

c     end subroutine DCRGU_Init
      end

c     ************************************************************

      subroutine DCRGU_CalcRate(c,T,r)
c     here you can specify your own rate laws for user defined
c     global gas-phase reactions
c     input:
c           c(*)  - concentration vector (according to DCS) [mol/m^3]
c           T     - temperature                             [K]
c     output:
c           r(*)  - reaction rates for all user defined
c                   global reactions (referenced by userID) [mol/m^3 s]
      implicit none
      include 'DCRGU_UserData.F'
      double precision c(*),T,r(*)
c
c     example:
c     assuming you have defined the following reaction in your mechanism:
c     <GLOBAL>
c        2 NO + O2 > 2 NO2
c        userID=1
c     </GLOBAL>
c
c     then provide code like this:
      r(1)=1.e5*c(no)*c(o2)*exp(-1000./T)
c


c     end subroutine DCRGU_CalcRate
      end

c     ************************************************************
```

Inhibition factors

The rate laws of global net reactions can be quite complex. However for technical systems like in the automotive exhaust-gas aftertreatment, a detailed reaction mechanism is either not available or not feasible for a fast solution. In such cases, experimental data is used to fit parameters that cover the desired range of conditions. Often, a simple Arrhenius law is not satisfying enough. Especially in cases where washcoat effects are directly taken into account in the rate expression, so-called inhibition factors are frequently used. The type of inhibition factors that can be evaluated within a global reaction definition is a follows:

$$I = \frac{1}{T^{\beta} \cdot \left(1 + K_1 X_1^{a_1} X_2^{a_2} + K_2 X_3^{a_3} X_4^{a_4} + \ldots\right)^{b_1} \cdot \left(1 + K_3 X_5^{a_5} X_6^{a_6} + \ldots\right)^{b_2} \cdot \ldots} \tag{3.4}$$

with

$$K_i = A_i \cdot \exp\left(-\frac{E_i}{RT}\right) \tag{3.5}$$

The inhibition factors contain a temperature dependent factor and a product of composition dependencies. Each of those factors consists of a sum of terms that depend on the mole fractions of one or two species. The factors themselves can be raised to an arbitrary power. Furthermore, the K_i are Arrhenius type of rate factors, with and pre-expontial factor A_i and an activation energy E_i. Therefore, several parameters can be defined: an temperature exponent β, mole fraction exponents a_i, factor exponents b_i, and Arrhenius parameters A_i and E_i

The definition of an inhibition factor will be illustrated by an example. Suppose an inhibition factor

$$I_1 = \frac{1}{T^{1.05} \left(1 + 0.3 \cdot X_{\mathrm{CO}}^{0.75} X_{\mathrm{C_3H_6}}^{0.5}\right) \left(1 + 5 \exp\left(-\frac{5000\mathrm{K}}{T}\right) \cdot X_{\mathrm{NO}}\right)^2} \tag{3.6}$$

To define this inhibition factor insert the following text in the **<MECHANISM>** section of the input file (it must be directly in the hierarchy level of the **<MECHANISM>** tag, **not** inside the **<GASPHASE>** or **<SURFACE>** tags)

```
<INHIBITION-FACTOR>
  ID=1
  T-Exponent=1.05
  <FACTOR>
    <SUMMAND>
      A=0.3
      CO ** 0.75
      C3H6 ** 0.5
    </SUMMAND>
  </FACTOR>
  <FACTOR>
    <SUMMAND>
      A=5
      Ea/R = 5000
      NO
    </SUMMAND>
    exponent=2
  </FACTOR>
</INHIBITION-FACTOR>
```

The complete list of options of the **<INHIBITION-FACTOR>** tag is given in the following table:

ID=*i*	ID of inhibition factor
T-Exponent=*d*	temperature exponent (default: 1)
<**Factor**>	factor of the form (1+Ki*Xi+...)

with following options in the **<FACTOR>** tag

exponent=*d*	exponent of the factor (default: 1)
<**Summand**>	summand of the form K * X1**n1 * X2**n2

and in the **<SUMMAND>** tag

A=*d*	preexponential factor (default: 1)
Ea/J_mol=*d*	activation energy [J/mol] (default: 0)
Ea/kJ_mol=*d*	activation energy [kJ/mol] (default: 0)
Ea/R=*d*	activation energy / R [K] (default: 0)
(species name)	species name
****=*d*	species exponent

Each inhibition factor is identified by an ID number, which must be an integer number (1,2,...). The inhibition factor can then be used to modify the Arrhenius law of a global reaction. This is done by the optional parameter **INHIBITION-FACTOR** within the **<GLOBAL>** tags as shown in the example below.

```
<GLOBAL>

  2 CO + O2 > 2 CO2

  <ARRHENIUS>
     A/cm_units = 1E10
     Ea/kJ_mol = 275
  </ARRHENIUS>
  INHIBITION-FACTOR=1

</GLOBAL>
```

3.8 Surface models

Note for DETCHEM Version 2.0 users: Because of some ambiguities between washcoat input and **<CHEMSURF>** input the data has been rearranged. For any simulation involving surface species the new section **<SURFACE-MODEL>** has to be defined. The information must be given between the **<MECHANISM>** section and the beginning of the application specific section. Now, as a consequence, all DETCHEM applications can apply at least the washcoat effectivness factor model.

The **<SURFACE-MODEL>** section contains information for the calculation of the net surface fluxes due to chemical reactions. Three properties influence the solution method:

- The steady-state solver **<CHEMSURF>**.
- The ratio of catalytic to geometric surface area.
- The washcoat diffusion model.

A simple example is shown below:

```
<SURFACE-MODEL>

  <CHEMSURF>
     time=1.         # integration time for surface coverages
     analytical=no   # use analytical Jacobian matrix
  </CHEMSURF>

  Fcatgeo=1.         #ratio of catalytic to geometric area

  # mo washcoat diffusion model defined
</SURFACE-MODEL>
```

The possible options inside the **<SURFACE-MODEL>** tags are listed in the following table:

<**CHEMSURF**>	CHEMSURF parameters
Fcatgeo=*d*	ratio catalytic / geometric surface (default: 1)
<**Fcatgeo**>	ratio catalytic /geometric surface (individual values for each surface type)
Ageo=*d*	geometric surface area (without washcoat)
<**active_moles**>	moles of active catalyst
Eff_Model=*s*	species for effictiveness factor model (skip this option for detailed washcoat model)
<**EFF_MODEL**>	list of species for effictiveness factor model (skip this option for detailed washcoat model)
<**MOLECULAR_DIFF**>	molecular diffusion model
<**KNUDSEN_DIFF**>	Knudsen diffusion model
<**MIXED_DIFF**>	molecular diffusion model
<**RANDOM_PORE**>	molecular diffusion model
<**OUTS**>	modified species list in OUTS file

3.8.1 <CHEMSURF> input

The DETCHEM library provides routines to calculate surface reaction source terms. For most DETCHEM applications steady-state conditions of the surface conditions are of interest. For this purpose, the routine **CHEMSURF** is called. **CHEMSURF** integrates the surface coverages under the constraints of a given gas-phase composition and temperature for a given time. It is assumed that this integration time is sufficient to reach steady state.

In many cases it is sufficient to specify an integration time only. To do so, use the **time** member. The integration time should be long enough in order to equilibrate the system. Use shorter times only in cases when transient effects of surface coverages are desired in your simulation (e. g. long term storage effects). On the other hand, some ill-conditioned surface reaction mechanisms may result in poor convergence behavior. In this case, long integration times may slow down the simulation.

Using the member **analytical** the user can choose if the Jacobian of the ODE system shall be determined analytically or numerically. In general, the analytical solution should be faster, but in some cases the numerical solution is more stable.

In case the ChemSurf integration process shall be monitored, the option **write_file** can be activated. Then, a file *chemsurf.plt* will be written upon the first call of ChemSurf during execution of any DETCHEM application, e. g. DETCHEM$^{\text{SURFPROBE}}$. This file contains the surface concentrations of all non-inert surface species vs. time. Note: When using the detailed reaction-diffusion washcoat model, this option will be ignored, because a different solver is called.

All options in the **<CHEMSURF>** section are summarized below.

time=*d*	integration time [s] (default: 1)
hini=*d*	initial integration time step [s] (default: 1e-8)
rTol=*d*	relative tolerance of surface coverages (default: 1e-4)
aTol=*d*	absoulute tolerance of surface coverages (default: 1e-20)
maxIter=*i*	maximum number of iterations (default: unlimited)
analytical=*b*	analytical (yes) or numerical (no) calculation of Jacobian (default: no)
write_file=*b*	write file *chemsurf.plt* during first ChemSurf call (default: no)

3.8.2 Catalytic vs. geometric surface area

The rates of reaction (in moles per area) are always defined for an ideal flat surface. Unless the surface is covered with a thin foil of the catalytic material, the boundary between the reactor volume and the surface is rough. Often, washcoats are used to increase the roughness and therefore to increase the surface area. To

account for this effect DETCHEM uses a factor $F_{\mathrm{cat/geo}}$ that gives the ratio of the catalytic and the geometric surface area.

The catalytic area of the reactor can be determined experimentally, e. g. by temperature programmed desoprtion experiments. If the number of active moles of catalyst is known, then this can be converted into an area using the surface site density Γ defined in the **<SPECIES>** section. The geometric area is the surface area of the reactor with a flat surface, i. e. in case of a cylindrical channel its equals its circumference times the channel length.

There are three ways defining the ratio:

- In case there is only one type of surface, it is sufficient to define a constant value.

```
Fcatgeo = 5.    # increase catalytic surface area by factor 5
```

- If you have specified more than one surface type in the **<SPECIES>** section of your input file, you can assign an individual ratio of catalytic area to geometric area for each of these surfce types.

```
<Fcatgeo>       # surface type dependend Fcat/geo
   Pt  0.75
   Rh  0.25
</Fcatgeo>
```

- Alternatively, the conversion of active moles to $F_{\mathrm{cat/geo}}$ can be done automatically.

```
<active_moles>
   Pt  0.002
   Rh  0.0005
</active_moles>
Ageo = 3.14e-4   # geometric surface area [m2]
```

3.8.3 Washcoat diffusion models

Washcoats are used to enlarge the catalytic surface of a reactor. However, because of their porous structure, diffusion limitations can become significant. Therefore, additional information about the geometric properties of the washcoat is required. An example of washcoat input is shown below.

```
<RANDOM_PORE>
 <ZONE>
  thickness=1.d-04    # thickness of the washcoat layer
  ngrid=10            # number of grid cells
  aspect=1            # aspect ratio of neighboring grid cells
 </ZONE>
 tau=3                # tortuosity

 <MACRO>
   porosity=0.15      # porosity
   diameter=1d-6      # pore diameter
 </MACRO>

 <MICRO>
   porosity=0.35
   diameter=1d-9
 </MICRO>

 <SOLVER>
   time=10            # solver integration time
   hini=1.d-4         # initial integration step size
 </SOLVER>

</RANDOM_PORE>
```

The fours diffusion models described in chapter 2.3.4 can be accessed by the following tags:

- **<MOLECULAR_DIFF>**
- **<KNUDSEN_DIFF>**
- **<MIXED_DIFF>**
- **<RANDOM_PORE>**

Including one of the fours tags in the **<SURFACE-MODEL>** section activates the (detailed) washcoat model. Note: In case the washcoat model shall be easily activated and deactivated, it is reccommended to put the washcoat definition into a separate file, e.ġ. ***washcoat.inp***, and activate or deactivate the include statement.

```
<SURFACE-MODEL>
  ...
  {include washcoat1.inp}    # activated washcoat diffusion model
# {include washcoat2.inp}    # deactivated washcoat diffusion model
</SURFACE-MODEL>
```

The options for each model are summarized below

<**ZONE**>	1-dimensional washcoat grid zone
porosity=*d*	porosity of washcoat
tau=*d*	tortuosity of washcoat
diameter=*d*	mean pore diameter [m]
<**MACRO**>	macro pore parameters (random model)
<**MICRO**>	micro pore parameters (random model)
relative_load=*d*	relative load of this zone
zone2:	start a second zone with different parameters
<**SOLVER**>	DAE solver parameters

The washcoat model basically solves a reaction diffusion equation. The tags **<MACRO>** and **<MICRO>** stand for macro pores and micro pores which are required for the random pore model. Otherwise, specify **porosity** and **diameter** for all cells once without these tags.

The information under the **<ZONE>** tag is used for the discretization of the governing equation 2.75. There can be any number of **<ZONE>** tags, however these zones will all have the same properties (except for the grid spacing). If the washcoat properties (porosity, pore diameter or loading) should vary along the grid, the zone definition can be devided into two layers by using the keyword **zone2:** as shown in the following example

```
<SURFACE-MODEL>
  <MIXED_DIFF>
    <ZONE>
      ngrid=10
      thickness=1.d-4
    </ZONE>
    porosity=.5
    tau=3
    diameter=1.-6
    relative_load= 1
  zone2:
    <ZONE>
      ngrid=10
      thickness=1.d-4
    </ZONE>
    porosity=.5
    tau=3
    diameter=1.-7
    relative_load= 3

    <SOLVER> time=10 </SOLVER>
  </MIXED_DIFF>
  ...
</SURFACE-MODEL>
```

In this example, the two zones are both .1 mm thick, but they differ in their pore diameters and their catalyst loadings. First, the fluid encouters a layer with large pores. Below there is a layer with smaller pores, but the specific catalytic surface area is three times larger. Use the parameter **relative_load** in order to define a ratio of the catalytic surface areas of the two layers (only the ratio of the two values is important). The actual amount of catalyst in the layers also depends on their thickness and the total catalytic surface area defined in **<Fcatgeo>**. Unfortunately it is not possible to define layers with different $F_{\mathrm{cat/geo}}$, i.e. the composition of the washcoat must not change. Otherwise the effectiveness factor model would not be applicable. Therefore it is not possible so far to define a system with two layers of different coatings, e.g. one layer platinum and one layer rhodium. However, by assigning **relative_load**=0 in one layer, an inactive layer can be created.

Options in **<ZONE>**:

ngrid=*i*	number of grid points
aspect=*d*	aspect ratio of adjacent grid cells (default: 1 ,i.e. equal spacing)
thickness=*d*	thickness of washcoat layer [m]

Options in **<MACRO>** and **<MICRO>**:

porosity=*d*	porosity of washcoat
diameter=*d*	pore diameter [m]

Options in **<SOLVER>**:

time=*d*	integration time [s] (default: 1)
hini=*d*	initial integration step size (default: 1e-8)
rTol=*d*	relative tolerances of concentrations and coverages (default: 1e-4)
aTol=*d*	absolute tolerances of concentrations and coverages (default: 1e-20)
maxiter=*i*	maximum number of iterations (default: no limit)

3.8.4 Effectiveness factor model

The detailed washcoat diffusion model is very time consuming and may not be available in some application (e. g. MONOLITH with transient coverages of a storage medium). In order to account for diffusion limitations of washcoats, an effectiveness factor approch can be applied.

In this model, the diffusion of one species is assumed to account for the diffusion limitions. Assign the name of one species to the **Eff_Model** or list some under **<EFF_MODEL>** in the **<SURFACE-MODEL>** section in addition to the washcaot definition.

Then no reaction-diffusion equation will be solved. Instead, the surface reaction rate will be calclulated the same way as without the washcoat diffusion model. Using the total rate for the chosen species, an effictiveness factor is calculated. Finally, all rates are multiplied by this effectiveness factor.

In case more than one species is listed, the effectiveness factor of the species that gives the strongest limitation will be selected automatically.

An example definition of an effectiveness factor model is shown below:

```
<SURFACE-MODEL>
  ...
  <MIXED_DIFF>
     <ZONE>
       thickness=1.d-04    # thickness of the washcoat layer
       ngrid=10            # dummy value (no grid generated)
     </ZONE>
     tau=3                 # tortuosity
     porosity=0.5          # porosity
     diameter=1d-9         # pore diameter

     <SOLVER>
       time=1              # dummy value (no separate integration)
     </SOLVER>

   </MIXED_DIFF>

   EFF_MODEL = CO          # CO is the diffusion limiting species

#  alternatively:
#  <EFF_MODEL>
#    CO   C3H6             # effectiveness factor for CO or C3H6 is applied
                           # (whichever is smaller)
#    threshold = 1e-10     # minimum mole fraction for Thiele modulus
#  </EFF_MODEL>

</SURFACE-MODEL>
```

Chapter 4

DETCHEM$^{\text{CHEMINP}}$

4.1 Introduction

DETCHEM$^{\text{CHEMINP}}$ is a pre-processor and translator for DETCHEM species and mechanism information files. It creates a compact species and mechanism input file ***chem.new*** as well as input files in CHEMKIN format.

4.2 User Input

DETCHEM$^{\text{CHEMINP}}$ requires one input file named ***cheminp.inp***. This file acts as a container for the species and mechanism information. If this information is stored in external files, e. g. ***species.inp*** and ***mech.inp***, then ***cheminp.inp*** may simply look like the following example.

```
<CHEMINP>
  {include species.inp}
  {include mech.inp}
</CHEMINP>
```

Like in all other DETCHEM applications, the files ***thermdata*** and ***moldata*** need to be present. For more information on these files please refer to chapter 3

4.3 Output

DETCHEM$^{\text{CHEMINP}}$ produces several output files.

The file ***chem.new*** contains the species and the mechanism information in a compact form. It can be used to replace the species and the mechanism information in any detchem apllication. Just write

```
{include chem.new}
```

at the beginning of the application specific input file.

The files ***thermdata.new*** and ***moldata.new*** contain the thermodynamic data and the molecular properties (respectively) of the required species. That is, only the information of the relavant species is extracted from ***thermdata*** and ***moldata***. If you want to use the new files with your application, please remove ***.new*** from the file names.

Furthermore, two files named ***gasmech.chemkin*** and ***surfmech.chemkin*** are created. These files contain the gas-phase and surface mechanisms in CHEMKIN format. Thus, DETCHEM$^{\text{CHEMINP}}$ can be used as a DETCHEM to CHEMKIN translator for input files.

4.4 Isotope analysis

An experimental method to examine reaction paths is the isotope analysis. Reactants could be marked using different isotopes in order to find out which product is formed from which reactant. This set-up can be simulated in any DETCHEM application. However, one needs to create new input files, because isotopes need to be treated as different species. Such new input files can be created using DETCHEM$^{\text{CHEMINP}}$.

This is done by adding a line to ***cheminp.inp*** using the option **`isotopes`** followed by the symbol of a chemical element. If you want to distinguish two oxigen isotopes, then use an input like the following:

```
<CHEMINP>
  {include species.inp}
  {include mech.inp}
  isotopes O
</CHEMINP>
```

After running DETCHEM^{CHEMINP} the files ***chem.new***, ***thermdata.new*** and ***moldata.new*** will contain species new species and reactions where the number of atoms that have been replaced by generic isotopes go through all possible combinations. (Note: It is recommended to use only species that contain no more than two atoms of the desired isotopes. The rate laws of complex reactions can only be translated if the desired atom does not appear more than once.) A new atom type, e. g. **0***, will be created. For each species containing a replaced atom, a generic species name will be created starting with an integer number that indicates the number of replace atoms. Therefore, the species **02** will be replaced by three species **02**, **102**, **202**.

The files ***gasmech.chemkin*** and ***surfmech.chemkin*** are not influenced by the isotope variation.

In order to use the isotope analysis in any application, copy the files ***chem.new***, ***thermdata.new*** and ***moldata.new*** to your working directory. Remove ***.new*** from ***thermdata.new*** and ***moldata.new***. Include ***chem.new*** at top of your application specific input file. Then, for example, you want to find out whether the oxygen in your product H_2O originates from O_2 or CO_2. In this case you may use the species **02** (unmarked) and **2C02** (marked) in your list of reactants. By looking at the product composition (**H20** and/or **1H20**), the reaction path can be determined.

4.5 Running the tool

The example directory is supplied with a ***go*** script which can be used to run the tool just by typing ***./go*** in the command prompt, or by any of the appropriate methods explained in the installation guide.

Chapter 5

DETCHEM^{GASPROBE}

5.1 Introduction

DETCHEM^{GASPROBE} is basically a tool to probe the gas-phase chemistry, thermodynamic and transport properties. Whenever a new gas-phase mechanism is to be used with the DETCHEM reactive flow models, it is recommended to test the mechanism using DETCHEM^{GASPROBE}

5.2 User Input

Before running DETCHEM^{GASPROBE} the user must prepare the input file ***gasprobe.inp***. An example is shown below.

```
{include species.inp}
{include mech.inp}

<GASPROBE>
  T/K=1200                  # operating temperature in K
  p/Pa=100000               # operating pressure in Pa
  <MOLEFRAC>
      NO2  0.1              # name and mole fraction
      N2   *                # balance mole fractions
  </MOLEFRAC>
</GASPROBE>
```

The file ***gasprobe.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition the files ***thermdata*** and ***moldata*** must be located in the executing directory. Alternatively, the pre-processed input file ***detchem.inp*** can be included.

The options within the **<GASPROBE>** tag are as follows.

T/K=*d*	temperature [K]
p/Pa=*d*	pressure [Pa]
<**MOLEFRAC**>	species mole fractions
<**MASSFRAC**>	species mass fractions

In the **<MASSFRAC>** or **<MOLEFRAC>** section, one species can be assigned a ***** instead of a numerical value. The mass or mole fraction of this species will be balanced to unity, respectively.

5.3 Output

DETCHEM^{GASPROBE} produces only screen output. The output contains information on operating conditions, mass and mole fractions, chemical source terms, heat capacity, enthalpy and entropy.

```
**************************************************************************
****                        DETCHEM_GASPROBE                          ****
****                   S. Tischer, O. Deutschmann                     ****
****                    VERSION 2.0 ** 04/06/22                       ****
**************************************************************************

 initial values
 Temperature [K]:   1200.
 Pressure   [Pa]:   100000.
 species      mass fraction       mole fraction
```

```
NO              0.00000000          0.00000000
O2              0.00000000          0.00000000
NO2             0.15432169          0.10000000
O               0.00000000          0.00000000
N2              0.84567831          0.90000000

 species      chemical source term [mol/(m^3 s)]
NO                0.000000E+00
O2                8.896840E+05
NO2              -8.896840E+05
O                 0.000000E+00
N2                4.448420E+05

 species     heat capacity [J/kg K]   enthalpy [J/kg]   entropy [J/kg K]
NO            1.16141735E+03         3.98023574E+06         1.16141735E+03
O2            1.11232116E+03         9.30481572E+05         1.11232116E+03
NO2           1.16560234E+03         1.65341201E+06         1.16560234E+03
O             1.30519414E+03         1.67661108E+07         1.30519414E+03
N2            1.20328747E+03         1.00382402E+06         1.20328747E+03
 average      1.19747184E+03         1.10406954E+06         1.28813565E+03

 species    mix diff coeff [m^2/s] viscosity [kg/m s]   heat cond [J/m K s]
NO            2.13229567E-04         4.83696341E-05         7.94895416E-02
O2            2.17922412E-04         5.39862825E-05         8.48086050E-02
NO2           1.66925845E-04         4.76563231E-05         7.58873612E-02
O             3.35261372E-04         6.30643181E-05         1.22901973E-01
N2            2.74149764E-04         4.67349007E-05         7.97217804E-02
 average                             4.68262397E-05         7.93296638E-02
```

5.4 Examples

There is an example supplied with the installation. The example directory contains the following files ***gasprobe.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism ***noo2m***, ***thermdata***, ***moldata***, and a script file ***go***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable DETCHEM_DIR that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

5.5 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.

5.6 Running the tool

The example directory is supplied with a ***go*** script which can be used to run the tool just by typing ***go*** in the command prompt, or by any of the appropriate methods explained in the installation guide.

Chapter 6

DETCHEM$^{\text{SURFPROBE}}$

6.1 Introduction

DETCHEM$^{\text{SURFPROBE}}$ is basically a tool to probe the surface chemistry. It calculates the surface coverages and fluxes at catalytic surfaces at constant ambient conditions. It is recommended to test a new mechanism using DETCHEM$^{\text{SURFPROBE}}$ before using it with DETCHEM reactive flow models. Most surface reactions are supposed to reach a steady state in few seconds or even less. DETCHEM$^{\text{SURFPROBE}}$ integrates the surface coverages until it reaches steady state.

6.2 User Input

Before running DETCHEM$^{\text{SURFPROBE}}$ the user must prepare the input file ***surfprobe.inp***.An example is shown below.

```
{include species.inp}
{include mech.inp}

<SURFACE-MODEL>
  <CHEMSURF>
      time=100              # integration time for surface chemistry
  </CHEMSURF>
</SURFACE-MODEL>

<SURFPROBE>
  T/K=1200                  # operating temperature in K
  p/Pa=1.d5                 # operating pressure in Pa

  <GASPHASE>
    <MOLEFRAC>
      NO2        0.1        # name and mole fraction
      N2         *          # balance mole fractions
    </MOLEFRAC>
  </GASPHASE>
</SURFPROBE>
```

The file ***surfprobe.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition the files ***thermdata*** and ***moldata*** must be located in the executing directory. Alternatively, the pre-processed input file ***detchem.inp*** can be included.

The options within the **<SURFPROBE>** tag are as follows.

Fcat/geo=*d*	ratio catalytic / geometric surface area
T/K=*d*	temperature [K]
p/Pa=*d*	pressure [Pa]
<**GASPHASE**>	gas-phase composition
<**SURFACE**>	surface coverages (default: initial coverages from species definition)

In the **<MASSFRAC>** or **<MOLEFRAC>** subsection, one species can be assigned a ***** instead of a numerical value. The mass or mole fraction of this species will be balanced to unity, respectively.

6.3 Output

DETCHEM$^{\text{SURFPROBE}}$ produces only screen output. The output contains information on operating conditions, mass and mole fractions, concentrations, chemical source terms, coverages and surface fluxes. It can easily be

inferred from the output, whether the surface mechanism reaches a steady state in the specified integration time.

```
*************************************************************************
****                      DETCHEM SURFPROBE                          ****
****    Vinod M Janardhanan, Steffen Tischer & Olaf Deutschmann      ****
****                    VERSION 2.0 ** 2004/06/22                    ****
*************************************************************************

 Initial Values
 Time of integration (s) :   0.
 Cat/Geom Surf Area      :   1.
 Temperature (K)         :   1200.
 Pressure (Pa)           :   100000.
 Gas Species     Mass fraction       Mole fraction     concentration (mol/m**3)
 NO              0.00000000          0.00000000          0.00000000
 O2              0.00000000          0.00000000          0.00000000
 NO2             0.15432169          0.10000000          1.00226377
 O               0.00000000          0.00000000          0.00000000
 N2              0.84567831          0.90000000          9.02037391
 Initial Coverage
 PT(s)      :    1.
 O(s)       :    0.
 NO2(s)     :    0.
 NO(s)      :    0.
 Coverage after   100.s:
 PT(s)      :    0.445116025
 O(s)       :    0.554778335
 NO2(s)     :    0.000105502945
 NO(s)      :    1.34432106E-07
 Fluxes at surface:    mole/(m**2 s)           kg/(m**2 s)
 NO                     0.44215373E+01          0.13267707E+00
 O2                     0.22107687E+01          0.70742386E-01
 NO2                   -0.44215373E+01         -0.20342167E+00
 O                      0.00000000E+00          0.00000000E+00
 N2                     0.00000000E+00          0.00000000E+00

 Net surface mass flux :  -2.21076882E-06  kg/(m**2 s)

 d(Coverage)/dt after   100.s:
 PT(s)                 -0.44304429E-06
 O(s)                   0.44206469E-06
 NO2(s)                -0.76278853E-07
 NO(s)                  0.76768657E-07
```

6.4 Examples

There is one example supplied with the installation. The example directory contains the following files ***surfprobe.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism ***noo2m***, a surface mechanism ***noo2sm***, ***thermdata***, ***moldata***, and a script file ***go***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable DETCHEM_DIR that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

6.5 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory which becomes your working directory and must contain all the input files mentioned in the above section. The user can copy the files from the example directory to the working directory and make the necessary changes.

6.6 Running the tool

The example directory is supplied with a ***go*** script which can be used to run the tool just by typing ***go*** in the command prompt, or by any of the appropriate methods explained in the installation guide.

Chapter 7

DETCHEM$^{\text{EQUIL}}$

7.1 Introduction

This code calculates the equilibrium composition of a given gas-phase mixture, which is defined by its thermodynamic potentials.

7.1.1 Theoretical background

The state of a closed system can be described by pressure p, temperature T, and the number of moles of each species n_i. This state can be expressed in terms of the Gibbs Free Enthapy as potential function $G(T, p, n_i)$. The system is in equilibrium, when G is a minimum with respect to all possible combinations of n_i.

In chemical systems, the mass is conserved, and hence the number of atoms must not change in a chemical reaction. If we denote the number of atoms of type j in species i by a_{ji}, then the changes δn_i of amount of species i must fulfil

$$\sum_i a_{ji}\,\delta n_i = 0 \quad \forall j. \tag{7.1}$$

If the system is described in terms of chemical reactions, we denote the stoichiometric coefficient of species i in reaction k by ν_{ik} and the reaction coordinate by ξ_k. Applying this, we can write

$$\delta n_i = \sum_k \nu_{ik}\xi_k \quad . \tag{7.2}$$

Inserting eq. 7.2 into eq. 7.1, we get

$$\sum_k \left(\sum_i a_{ji}\nu_{ik} \right) \xi_k = 0 \tag{7.3}$$

This must be fulfilled for arbitrary ξ_k. Therefore the inner brackets must vanish. The matrices $(a_{ji})^{\mathrm{T}}$ and (ν_{ik}) are othogonal with respect to each other. Hence, for any (ν_{ik}) with maximum rank any possible reaction can be written as linear combination of its columns. The columns of matrix (ν_{ik}) are the stoichiometric coefficients of hypothetical reactions.

Instead of solving the minimization problem with constraints (eq. 7.1), we can solve the minimization problem for the hypothetical reactions.

For a mixture of ideal gases, the Gibbs Free Enthalpy is given by the expression

$$G(T, p, n_i) = \sum_i n_i \left(g_i^0(T) + RT \ln \frac{p_i}{p^0} \right) \tag{7.4}$$

where g_i^0 denotes the standard specific molar free enthalpy at pressure p^0. p_i is the partial pressure according to the ideal gas law

$$p_i = \frac{n_i RT}{V} \quad . \tag{7.5}$$

Equation 7.4 can be generalized to other mixtures than ideal gases by introduction of an activity a_i that replaces the term $\frac{p_i}{p^0}$. Here, we also consider mixtures of ideal liquids, where a_i is the mole fraction of species i in the condensed phase.

Reaction k is in equilibrium when

$$\left(\frac{\partial G(T,p,n_i+\nu_{ik}\xi_k)}{\partial \xi_k}\right)_{T,p} = 0 \quad . \tag{7.6}$$

This is equivalent to the formualtion involving the equilibrium constant

$$K_p = \exp\left(\frac{-\Delta_R G^0}{RT}\right) = \prod_i (a_i)^{\nu_{ik}} \tag{7.7}$$

with

$$\Delta_R G^0 = \sum_i \nu_{ik} g_i^0 \quad . \tag{7.8}$$

7.1.2 Algorithm

Applying the mathematical model, we can outline the following algorithm:

1. Construct hypothetical reactions preferring those involving species with large amounts,
2. Equilibrate all reactions sequentially,
3. If amounts of all species have changed less than a given tolerance, then finish, else go to 1.

7.2 User input

DETCHEM^{EQUIL} requires an input file ***equil.inp***, thermodynamic data in file ***thermdata***, and the database for molecular properties ***moldata***. For convenience, parts of the input file ***equil.inp***, e. g. the species list, may be saved in a seperate file and included using an **`include`** statement.

7.2.1 *equil.inp*

For running DETCHEM^{EQUIL} once use an input file like this:

```
<EQUIL>
  <SPECIES>
    H2  O2  OH  H2O H O H2O2 CH4 CO2 CO N2
  </SPECIES>
  <CONDENSED>          # optional list of species in a condensed phase
    C  2260            # species name and its density in kg/m3
  </CONDENSED>

  T=1000               # temperature in K
  const p T            # constraints: constant pressure & temperature

  <MOLEFRAC>           # initial composition
    CH4 0.15
    O2  0.15
    N2  *
  </MOLEFRAC>

  atol=1.d-20          # absoulute tolerances
  rtol=1.d-6           # relative tolerances
  Ttol=0.1             # temperature tolerance
</EQUIL>
```

Instead of a single temperature, the user can also specify a temperature range by replacing the **`T`** option with a definition in the following format:

```
  <T> from 500 to 1200 step 50 </T>
```

The table below lists all available options. Where applicable, the default values are given.

<**SPECIES**>	species list
<**CONDENSED**>	list of species in a condensed phase (optional)
T=*d*	temperature [K] (default: 298)
<**T**>	list of temperatures [K]
p=*d*	pressure [Pa] (default: 101325)
V=*d*	volume [m^3] (default: 1e-3)
<**MOLEFRAC**>	species mole fractions
<**MASSFRAC**>	species mass fractions
atol=*d*	absolute tolerance of concentrations [mol/m^3] (default: 1e-30)
rtol=*d*	realtive tolerance of concentrations (default: 1e-6)
Ttol=*d*	absolute tolerance of temperature [K] (default: 0.1)
const v_1 v_2	operation condition: choose 2 constant properties from p, V, T, S, H

Two constraints can be chosen from constant pressure **p**, volume **V**, temperature **T**, enthalpy **H** or entropy **S**. However, since enthalpy is a function of temperature only for ideal mixtures, constants **T** and **H** cannot be selected at the same time. The classical equilibrium problem is solved for **const T p**. If you want the adiabatic case choose **const H p**.

7.2.2 *thermdata*

For a detailed description of the ***thermdata*** file see chapter 3.2.

Here it shall only be mentioned that besides the thermodynamic coefficients the file ***thermdata*** also contains information about the atomic composition of each species. This information plays an important role when running DETCHEM$^{\text{EQUIL}}$. Make sure that the first line of each species contains the atomic composition in the correct format, e.g.

```
CH3O                  C   1H   3O   1    0G   300.000  5000.000             1
                      ^    ^    ^    ^
                      C    H3   O    none
```

The symbols are expected in colums 25, 30, 35, and 40. Each atom symbol can be up to two characters followed by an integer value in the next 3 columns.

7.3 Output

7.3.1 Screen output

After successfully executing DETCHEM$^{\text{EQUIL}}$ the screen will look like this:

```
*************************************************************************
****                        DETCHEM EQUIL                            ****
****                          S. Tischer                             ****
****                  VERSION 2.0.100 ** 05/04/27                    ****
*************************************************************************
 species:
 H2        H   2
 O2        O   2
 OH        H   1 O   1
 H2O       H   2 O   1
 H         H   1
 O         O   1
 H2O2      H   2 O   2
 CH4       H   4 C   1
 CO2       O   2 C   1
 CO        O   1 C   1
 N2        N   2
 initial conditions:
 p=  101325. Pa
 V=  0.001 m^3
 T=  500. K
 S=  5.25471088 J/K
 U= -221.736483 J
 H= -120.411483 J
 F= -2849.09192 J
 G= -2747.76692 J
 species          moles      mole frac
H2         0.000000E+00   0.000000E+00
O2         3.655958E-03   1.500000E-01
OH         0.000000E+00   0.000000E+00
H2O        0.000000E+00   0.000000E+00
H          0.000000E+00   0.000000E+00
O          0.000000E+00   0.000000E+00
H2O2       0.000000E+00   0.000000E+00
```

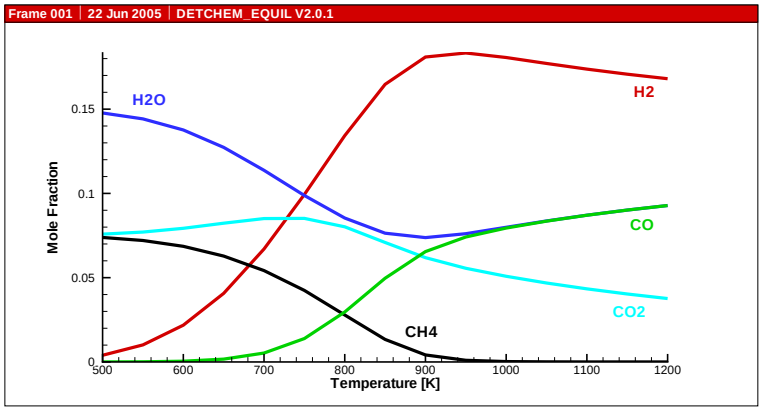
```
CH4         3.655958E-03   1.500000E-01
CO2         0.000000E+00   0.000000E+00
CO          0.000000E+00   0.000000E+00
N2          1.706114E-02   7.000000E-01
 hypothetical reactions:
 2 H2O       1 CH4       >  4 H2        1 CO2
 2 H2O       1 CO2       >  2 O2        1 CH4
 6 H2O       1 CO2       >  8 OH        1 CH4
 2 H2O       1 CH4       >  8 H         1 CO2
 2 H2O       1 CO2       >  4 O         1 CH4
 6 H2O       1 CO2       >  4 H2O2      1 CH4
 1 CH4       3 CO2       >  2 H2O       4 CO
 Thermodynamic equilibrium:
 p=  101325. Pa
 V=  0.00100199314 m^3
 T=  500. K
 S=  5.28418255 J/K
 U= -1681.16951 J
 H= -1579.64255 J
 F= -4323.26079 J
 G= -4221.73383 J
 species          moles      mole frac
H2          9.680003E-05   3.963701E-03
O2          7.432120E-23   3.043253E-21
OH          1.950599E-22   7.987177E-21
H2O         3.607736E-03   1.477271E-01
H           9.364217E-23   3.834395E-21
O           4.954746E-23   2.028835E-21
H2O2        4.409470E-23   1.805559E-21
CH4         1.803689E-03   7.385622E-02
CO2         1.851910E-03   7.583075E-02
CO          3.577961E-07   1.465079E-05
N2          1.706114E-02   6.986076E-01
```

First, the atomic compostion of all species is printed, in order to allow checking that the data is read correctly. Then the initial conditions are shown (thermodynamic variables and composition). As an additional information, one set of hypothetical reactions is printed. Finally, the thermodynamic variables and the compostion of the mixture in equilibrium are shown.

7.3.2 *equil.plt*

DETCHEM^{EQUIL} creates a file ***equil.plt*** that contains the equilibrium composition(s) and their thermodynamic variables. This file may be useful for graphical data evaluation in case of multi temperature runs. However, a new file is created upon each call of DETCHEM^{EQUIL}, so data will not be appended.

The output file is in TECPLOT format. TECPLOT will create a line graph upon loading the file ***equil.plt***. Nevertheless, this format also allows for easy import into spreadsheet programs like Microsoft Excel. The variables in the file are: initial temperature, equilibrium values of pressure, volume, temperature, entropy, enthalpy, free enthalpy, amount of species, and species mole fractions (separately for gas and condensed phase).



7.4 Example

The distribution contains an example for DETCHEM^{EQUIL} in the directory ***($DETCHEM_DIR)/EQUIL/example***. It contains three input files ***equil.inp***, ***thermdata***, and ***moldata***. The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

7.5 Setting up a problem

It is recommended to create new problems in separate directories. The user must supply the files ***equil.inp***, ***thermdata***, and ***moldata***. In most cases it might be convenient to copy an existing directory and modify the files.

If the same species data is used by more than one DETCHEM application, it might also be advisible to save the species information together with ***thermdata*** and ***moldata*** in a separate directory and create symbolic links to your working directory, e.g.

```
ln -s ~/detchem/data/species/thermdata .
```

Make sure that all species in your species list are also contained in ***thermdata*** and ***moldata***. Define the initial conditions and tolerances and start the program.

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***equil***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/equil*** and call ***./equil***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 8

DETCHEM$^{\mathrm{BATCH}}$

8.1 Introduction

The batch code simulates homogeneous gas-phase and heterogeneous surface reactions in a batch reactor. The user can specify his own time-dependent pressure, volume and temperature profiles. In case of missing temperature profile, isenthalpic or adiabatic analysis can be performed.

8.1.1 Governing equations

$$\frac{dn_k}{dt} = V\dot{\omega}_k + A\dot{s}_k \quad \text{gas-phase species} \tag{8.1}$$

$$\frac{dn_k}{dt} = \alpha A\dot{s}_k \quad \text{surface species} \tag{8.2}$$

$$pV = nRT \quad \text{ideal gas equation} \tag{8.3}$$

In the above equations $\dot{\omega}_k$, $\dot{s}_k$, A, V, α represent the gas-phase reaction rate, surface reaction rate, catalytic surface area, reactor volume, and surface relaxation factor respectively. The species concentrations n_k are expressed in terms of mole numbers.

If the reactor temperature as a function of time is not known the user can define an isenthalpic or adiabatic reactor:

$$\frac{dH}{dt} = 0 \quad \text{in the isenthalpic case} \tag{8.4}$$

$$\frac{dQ}{dt} = 0 \quad \text{in the adiabatic case} \tag{8.5}$$

The expressions (8.4) and (8.5) provide an additional equation for the reactor temperature.

DETCHEM$^{\mathrm{BATCH}}$ can perform sensitivity analysis for all species with respect to all or reduced number of kinetic parameters. Pre-exponential factors A_j ($A_{0,j}$ and $A_{\infty,j}$ in case of Troe reactions) and initial sticking coefficients S_j^0 are considered parameters in DETCHEM$^{\mathrm{BATCH}}$. Absolute and relative sensitivity values are calculated using the following expressions:

$$E_{i,j} = \frac{dn_i}{dp_j} \quad \text{absolute sensitivities} \tag{8.6}$$

$$E_{i,j}^{rel} = \frac{p_j}{n_i}\frac{\partial n_i}{\partial p_j} = \frac{\partial \ln n_i}{\partial \ln p_j} \quad \text{relative sensitivities} \tag{8.7}$$

In order to improve the accuracy of the computed relative sensitivities, an option for sensitivity analysis with respect to $\ln p_j$ is also introduced:

$$E_{i,j}^{\ln p_j} = p_j\frac{\partial n_i}{\partial p_j} = \frac{\partial n_i}{\partial \ln p_j} \tag{8.8}$$

8.1.2 Solution

In DETCHEM$^{\text{BATCH}}$ the standard implicit solvers LIMEX or DAESOL can be used to solve the coupled governing equations. The temperature solution in batch code offers user-defined time-dependent thermal conditions and non-isothermal isenthalpic or adiabatic conditions in a batch reactor.

8.2 User Input

Before running DETCHEM$^{\text{BATCH}}$ the user must prepare an input file ***batch.inp***. The program considers four batch reactor types:

1. with user-defined volume and temperature profiles
2. with user-defined pressure and temperature profiles
3. with user-defined volume profile (isenthalpic or adiabatic reactor)
4. with user-defined pressure profile (isenthalpic or adiabatic reactor)

Examples for these four cases are shown in the table below.

The file ***batch.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition, the files ***thermdata*** and ***moldata*** must be located in the executing directory. Alternatively, the pre-processed input file ***detchem.inp*** can be included.

example 1

```
{include species.inp}
{include mech.inp}

<BATCH>

  solver_id = 0   # 0 LIMEX, 1 DAESOL
  p/Pa = 8000.d0
  A/m2 = 1.0
  time = 100

  h    = 1.0e-08
  hmax = 1.d0
  rtol = 1.0e-02
  atol = 1.0e-09
  monitor = 1

  <MOLEFRAC>
     C3H6 0.996
     CH4  0.003
     C3H8 0.001
  </MOLEFRAC>

  <V_PROFILE>
    0.0   1.5
   30.0   1.0
   70.0   1.5
  100.0   1.0
  </V_PROFILE>

  <T_PROFILE>
    0.0   1173
   50.0   1273
  100.0   1173
  </T_PROFILE>

  <OUTPUT>
    mole          = y
    concentration = y
    mole_fraction = y
    mass_fraction = y
    fileNr = 1
    dt_out = 0.01
  </OUTPUT>

</BATCH>
```

example 2

```
{include species.inp}
{include mech.inp}

<BATCH>

  solver_id = 0   # 0 LIMEX, 1 DAESOL
  V/m3 = 1.0
  A/m2 = 1.0
  time = 100

  h    = 1.0e-08
  hmax = 1.d0
  rtol = 1.0e-02
  atol = 1.0e-09
  monitor = 1

  <MOLEFRAC>
     C3H6 0.996
     CH4  0.003
     C3H8 0.001
  </MOLEFRAC>

  <P_PROFILE>
    0.0    8000
   30.0   10000
   70.0    8000
  100.0   10000
  </P_PROFILE>

  <T_PROFILE>
    0.0   1173
   50.0   1273
  100.0   1173
  </T_PROFILE>

  <OUTPUT>
    mole          = y
    concentration = y
    mole_fraction = y
    mass_fraction = y
    fileNr = 1
    dt_out = 0.01
  </OUTPUT>

</BATCH>
```

example 3

```
{include species.inp}
{include mech.inp}

<BATCH>

  solver_id = 1   # 0 LIMEX, 1 DAESOL
  p/Pa = 8000.d0
  A/m2 = 1.0
  time = 100

  h    = 1.0e-08
  hmax = 1.d0
  rtol = 1.0e-03
  atol = 1.0e-09
  monitor = 1

  <MOLEFRAC>
     C3H6 0.996
     CH4  0.003
     C3H8 0.001
  </MOLEFRAC>

  <V_PROFILE>
    0.0   1.5
   30.0   1.0
   70.0   1.5
  100.0   1.0
  </V_PROFILE>

  <CONST_QUANTITY>
    T/K            = 1173
    const_quantity = Q      # Q or H
    surf_thermdata = n
  </CONST_QUANTITY>

  <OUTPUT>
    mole          = y
    concentration = y
    mole_fraction = y
    mass_fraction = y
    fileNr = 1
    dt_out = 0.01
  </OUTPUT>

</BATCH>
```

example 4

```
{include species.inp}
{include mech.inp}

<BATCH>

  solver_id = 1   # 0 LIMEX, 1 DAESOL
  V/m3 = 1.0
  A/m2 = 1.0
  time = 100

  h    = 1.0e-08
  hmax = 1.d0
  rtol = 1.0e-02
  atol = 1.0e-09
  monitor = 1

  <MOLEFRAC>
     C3H6 0.996
     CH4  0.003
     C3H8 0.001
  </MOLEFRAC>

  <P_PROFILE>
    0.0    8000
   30.0   10000
   70.0    8000
  100.0   10000
  </P_PROFILE>

  <CONST_QUANTITY>
    T/K            = 1173
    const_quantity = Q      # Q or H
    surf_thermdata = n
  </CONST_QUANTITY>

  <OUTPUT>
    mole          = y
    concentration = y
    mole_fraction = y
    mass_fraction = y
    fileNr = 1
    dt_out = 0.01
  </OUTPUT>

</BATCH>
```

The input file must always contain either user-specified pressure (exapmles 1 and 3) or volume (examples 2 and 4) profile, but not both profiles simultaneously. The user can also define a time-dependent temperature profile (examples 1 and 2) or specify adiabatic (examples 3 and 4) or isenthapic conditions. Initial values for pressure, volume and/or temperature must be specified, provided the corresponding parameter has no user-specified profile. This means that in case of user-defined volume profile, initial pressure is to be given (P/Pa in examples 1 and 3), in case of user-defined pressure profile, initial volume is to be given (V/m3 examples 2 and 4), and in the adiabatic (or isenthalpic) examples initial temperatures (T/K in the **<CONST_QUANTITY>** section in examples 3 and 4) have to be specified. The simultaneous presence of user-defined profile and initial value for the same parameter (p, V or T) is ambiguous and causes an error message.

Profiles are defined by pairs of values. The first value is the time, and the second one is the value of the corresponding parameter (p, V or T). Time must always increase, i.e., each time value must be greater than the preceding times. For instance:

```
<V_Profile>
   0.0   0.001
  20.0   0.002
  40.0   0.0025
  60.0   0.0028
 100.0   0.003
</V_Profile>
```

The user can specify an arbitrary number of pairs. If the profile consists only of one pair of values, the specified parameter (p, V or T) remains constant during the whole simulation. Intermediate states are determined by means of linear interpolation. For example: according to the volume profile shown above, the reactor volume after 30 s will be 0.00225 m^3.

Adiabatic and isenthalpic conditions are specified in the <**CONST_QUANTITY**> section. It contains the initial reactor temperature T/K and the options const_quantity and surf_thermdata. The const_quantity option can be set to Q for adiabatic and to H for isenthalpic simulation. For adiabatic-isobaric problems either of both values can be used. However, the user is recommended to use const_quantity=H in this case because the solution converges faster and provides more accurate results. The surf_thermdata determines whether thermodynamic data for the surface species have to be taken into account (y) or not (n). This option ist set to n by default because in most cases thermodynamic data for the surface species are not available.

The options within the **<BATCH>** tag are as follows:

solver_id=*i*	swtich between the LIMEX (0) and DAESOL (1) solvers (default: 0)
p/Pa=*d*	initial pressure [Pa]
V/m3=*d*	initial reactor volume [m^3]
A/m2=*d*	catalytic surface area [m^2]
time=*d*	integration time [s]
ini_surf=*b*	calculate equilibrium surface coverages for initialization (default: no). Implemented only with the LIMEX solver
cov_relax=*d*	under relaxation factor for surface coverages (default: 1)
atol=*d*	absolute tolerances for concentrations [mol/m^3] (default: 1e-12)
rtol=*d*	relative tolerances for concentrations (default: 1e-6)
h=*d*	initial integration step size [s] (default: 1e-10)
hmax=*d*	maximal integration step size [s] (default: not defined, the solver determines the step size)
daesol_ESW=*d*	tolerance needed by the DAESOL solver (default: 1.0d-02)
daesol_ETA=*d*	tolerance needed by the DAESOL solver (default: 1.0d-11)
<**MOLEFRAC**>	initial gas-phase mole fractions
<**MASSFRAC**>	initial gas-phase mass fractions
<**COVERAGE**>	initial surface coverages (default: coverages from <SPECIES> section)
<**V_Profile**>	user-specified time dependent volume profile
<**p_Profile**>	user-specified time dependent pressure profile
<**T_Profile**>	user-specified time dependent temperature profile
<**CONST_QUANTITY**>	section to define isenthalpic or adiabatic conditions
<**OUTPUT**>	output options
<**SENSITIVITY**>	sensitivity analysis (only with the DAESOL solver)

The tolerances daesol_ESW and daesol_ETA are needed by the the DAESOL solver. For simple gas-phase problems, default values work fine, but by stiff gas-phase and surface systems convergence difficulties may occur. In case of divergence or slow convergence, following values are recommended: daesol_ESW in the range 1E-2 → 1E-7 and daesol_ETA in the range 1E-10 → 1E-14.

The options within the **<OUTPUT>** tag are:

mole=*b*	output in terms of moles; output file: ***batch_n_***.plt*** (default: y)
concentration=*b*	output in terms of concentrations; output file: ***batch_c_***.plt*** (default: n)
mole_fraction=*b*	output in terms of mole fractions for gas-phase species and of coverages for surface species; output file: ***batch_x_***.plt*** (default: n)
mass_fraction=*b*	output in terms of mass fractions for gas-phase species and of coverages for surface species; output file: ***batch_y_***.plt*** (default: n)
fileNr=*i*	output file number (default: 1)
dt_out=*d*	minimum step size for output [s] (default: none, solution is written at every step)
monitor=*i*	0 - no time output on the screen, 1 - time output on the screen (default: 0)

The options within the **<CONST_QUANTITY>** tag are:

T/K=*d*	initial temperature [K]
const_quantity	H for isenthalpic problems, Q for adiabatic problems (default: H)
surf_thermdata=*b*	thermodynamic data for the surface species (default: n)

The options within the **<SENSITIVITY>** tag are:

complete=*b*	Sensitivity analysis with respect to all kinetic parameters (default: n)
absolute_sensitivity=*b*	output of absolute sensitivities; output file: ***batch_sens_abs_***.plt*** (default: n)
relative_sensitivity=*b*	output of relative sensitivities; output file: ***batch_sens_rel_***.plt*** (default: n)
d(n_i)/d(lnp_j)=*b*	computation of sensitivity values with respect to $\ln p_j$ (eq. 8.8); (default: y)
threshold_lnp=*d*	computation of sensitivity values with respect to $\ln p_j$ (eq. 8.8) only for $p_j >$ **threshold_lnp**; (in force only when **d(n_i)/d(lnp_j)**=y; default: 1000)

The option complete in the **<SENSITIVITY>** tag determines with respect to which kinetic parameters sensitivity coefficients have to be computed. If it is set to yes sensitivities with respect to all reactions will be calculated. If complete=n, only flagged reactions in the mechanism file will be taken into account. The flag is the symbol (*) and it has to be put before the first kinetic parameter (column 45). In the example below, only the pre-exponential factor of the first reaction will be included in the sensitivity analysis.

```
CH      +H      =C      +H2             * 1.100E+14  0.0      0.000
3CH2    +C      =C2H    +H                0.500E+14  0.0      0.000
```

The way DETCHEM$^{\text{BATCH}}$ internally computes the sensitivity values is controlled by the option **d(n_i)/d(lnp_j)**. If it is set to (y), sensitivities $E_{i,j}^{\ln p_j}$ will be calculated, otherwise absolute sensitivities $E_{i,j}$ will be calculated. Relative sensitivities $E_{i,j}^{rel}$ are derived from $E_{i,j}^{\ln p_j}$ by multiplying by n_i and from $E_{i,j}^{p_j}$ by multiplying by n_i and dividing by p_j. The first way is the more accurate one from numerical point of view. Relative sensitivities are more useful than absolute sensitivities during mechnism development. For that reason, users are recommended to use **d(n_i)/d(lnp_j)**=y.

8.3 Output

8.3.1 Screen output

On screen the integration progress is monitored. The values in the column displayed stand for time (monitor = 1).

```
*************************************************************************
****                        DETCHEM BATCH                            ****
****                N. Mladenov, S. Tischer, O. Deutschmann          ****
****                  VERSION 2.1 ** 03/08/07                        ****
*************************************************************************
    time,s
    0.0000000E+00
    5.1100105E-02
    1.0197794E-01
    1.5368989E-01
    ............
    5.0000000E+00
 DETCHEM -- copyright by O. Deutschmann -- licence agreement required
 For further information visit www.detchem.com
BATCH finished successfully
```

8.3.2 *batch_***.plt*

DETCHEM$^{\text{BATCH}}$ produces four output files: ***batch_n_***.plt***, ***batch_c_***.plt***, ***batch_x_***.plt*** and ***batch_y_***.plt***. The output files are in TECPLOT format. TECPLOT will create a line graph upon loading a plt file. Nevertheless, this format also allows for easy import into spreadsheet programs like Microsoft Excel. The variables in the files are: time, pressure, volume, temperature and species data. Species data can be written as:

1. mole numbers - prefixed with n and written in ***batch_n_***.plt***
2. concentrations - prefixed with c and written in ***batch_c_***.plt***
3. mole fractions - prefixed with x and written in ***batch_x_***.plt*** (only gas-phase species)
4. mass fractions - prefixed with y and written in ***batch_y_***.plt*** (only gas-phase species)

5. coverages - prefixed with cov and written in ***batch_x_***.plt*** and ***batch_y_***.plt*** (only surface species)

The output of the four plt files is controlled via the options mole, concetration, mole_fraction and mass_fraction in the <**OUTPUT**> section. By default, only the ***batch_n_***.plt*** is written.

8.3.3 *batch_sens_***.txt*

Absolute sensitivity coefficients (eq. 8.6) are written to the file ***batch_sens_abs***.txt***, while relative sensitivities (eq. 8.7) are written to ***batch_sens_rel***.txt***. Both files have a similar structure. The first line contains one of the following three strings identifying the kind of the written sensitivity values:

- SENS_ij=d(N_i)/d(P_j) - for absolute sensitivities $E_{i,j}$ (in case **d(n_i)/d(lnp_j)**=n, only in ***batch_sens_abs***.txt***)
- SENS_ij=d(N_i)/d(lnP_j) - for sensitivities $E_{i,j}^{\ln p_j}$ (in case **d(n_i)/d(lnp_j)**=y, only in ***batch_sens_abs***.txt***)
- SENS_ij=d(lnN_i)/d(lnP_j) - for relative sensitivities $E_{i,j}^{rel}$ (only in ***batch_sens_rel***.txt***)

The variables' names for which sensitivity analysis has been done, are listed in a **<VARIABLES>** tag. Such variables could be not only the species' concentrations n_i but pressure P, volume V and temperature T of the reactor as well. It is followed by a **<REACTIONS>** tag, where the reactions and their corresponing parameters (in case **d(n_i)/d(lnp_j)**=n) or the natural logarithm of the parameters (in case **d(n_i)/d(lnp_j)**=y) used for the sensitivity analysis are listed. The tag also provides information about the kind of each reaction: gas-phase or surface; Arrhenius, Troe, global or reaction with given sticking coefficient. The **<SENSITIVITY_VALUES>** tag contains the sensitivity values ($E_{i,j}$, $E_{i,j}^{\ln p_j}$ or $E_{i,j}^{rel}$) computed during the integration time. Files ***batch_sens_abs***.txt*** additionally contain the tag **<SOLUTION_MAX_VALUES>** where maximal values for all variables for the whole integration time are stored.

8.4 Examples

There is one example supplied along with the installation. The example directory contains the following files: ***batch.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism ***noo2m***, ***thermdata***, ***moldata***, and a script file ***go***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

8.5 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.

8.6 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***batch***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/batch*** and call ***./batch***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 9

DETCHEM^{CSTR}

9.1 Introduction

The program DETCHEM^{CSTR} simulates the behavior of the continuous stirred tank reactor (CSTR) under different temperature conditions for applications involving gasses. The code is designed to simulate homogeneous gas-phase and heterogeneous surface reactions in a CSTR.

9.1.1 Governing equations

$$\frac{dc_k}{dt} = \dot{\omega}_k + \frac{A}{V}\dot{s}_k + \frac{\dot{V}_{\text{in}}}{V}c_{k,\text{in}} - \frac{\dot{V}_{\text{out}}}{V}c_k \quad \text{gas-phase species} \tag{9.1}$$

$$\frac{dc_k}{dt} = \alpha\dot{s}_k \quad \text{surface species} \tag{9.2}$$

$$\frac{dH}{dt} = \sum_k \left(h_k(T_{\text{in}})c_{k,\text{in}}\dot{V}_{\text{in}} - h_k(T)c_k\dot{V}_{\text{out}}\right) \tag{9.3}$$

$$\dot{V}_{\text{out}} = \frac{T_{\text{out}}\bar{M}_{\text{in}}}{T_{\text{in}}\bar{M}_{\text{out}}}\dot{V}_{\text{in}} \tag{9.4}$$

In the above equation $\dot{\omega}_k$, $\dot{s}_k$, A, V, $\dot{V}_{\text{in}}$, $\dot{V}_{\text{out}}$, α, T_{out}, T_{in}, $\bar{M}_{\text{in}}$, $\bar{M}_{\text{out}}$, h_{in}, and h represent the gas-phase reaction rate, surface reaction rate, surface area, reactor volume, inlet flow, outlet flow, coverage relaxation factor, outlet temperature, inlet temperature, mean inlet molar mass, mean outlet molar mass, enthalpy at inlet and enthalpy at outlet respectively.

9.1.2 Solution

In DETCHEM^{CSTR} the standard implicit code LIMEX is used to solve the coupled governing equations. The temperature solution in cstr code offers isothermal and non-isothermal adiabatic conditions in a continuous stirred tank reactor.

9.2 User Input

Before running DETCHEM^{CSTR} the user must prepare an input file ***cstr.inp***. An example is shown below.

```
{include species.inp}
{include mech.inp}
<SURFACE-MODEL>
  <CHEMSURF>
    time=1.          # suface integration time for steady state
  </CHEMSURF>
</SURFACE-MODEL>

<CSTR>
  <INITIAL>
    T = 1200         # reactor temperature in K
    <MOLEFRAC>
```

```
      NO2    0.1     # species name and mole fraction
      N2     *       # balance mole fraction
    </MOLEFRAC>
  </INITIAL>

  <INLET>
    T = 1200          # inlet temperature in K
    <MOLEFRAC>
      NO2   0.1       # inlet mole fraction
      N2    *
    </MOLEFRAC>
    Vflux = 0.01      # inlet flux in m^3/s
  </INLET>

  p/Pa= 100000        # pressure in Pa
  time=100            # integration time

  volume = 1          # reactor volume in m^3
  surface = 1         # catalytic surface in m^2

  isothermal=n        # isothermal=y: T=const,
                      # isothermal=n: solve enthalpy equation
  fileNr=1            # number of output file
  dt_out=0.05         # minimum output step size

  ini_surf=y          # find steady-state surface coverages  before
                      # integration starts
</CSTR>
```

The file ***cstr.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition the files ***thermdata*** and ***moldata*** must be located in the executing directory. Alternatively, the pre-processed input file ***detchem.inp*** can be included.

The options within the **<CSTR>** tag are as follows.

<**INITIAL**>	initial conditions
<**INLET**>	inlet conditions
p/Pa=*d*	pressure [Pa] (default: 100000)
p/bar=*d*	pressure [bar] (default: 1)
time=*d*	integration time [s]
volume=*d*	reactor volume [m^3] (default: 1)
surface=*d*	catalytic surface area [m^2]
isothermal=*b*	isothermal (yes) or adiabatic (no) conditions (default: no)
dT/s=*d*	linear temperature gradient (per second) for TPD simulations
dT/min=*d*	linear temperature gradient (per minute) for TPD simulations
cov_relax=*d*	under relaxation factor for surface coverages (default: 1)
ini_surf=*b*	call CHEMSURF for initial coverages (yes) or use default (no) (default: no)
fileNr=*i*	file number for output, e.g. cstr001.plt (default: 1)
dt_out=*d*	minimum step size for output [s] (default: none)
atol=*d*	absolute tolerances for concentrations [mol/m^3] (default: 1e-12)
rtol=*d*	relative tolerances for concentrations (default: 1e-6)
h=*d*	initial integration step size [s] (default: 1e-10)

The subsection **<INITIAL>** may contain:

T=*d*	temperature [K]
<**MOLEFRAC**>	gas-phase mole fractions
<**MASSFRAC**>	gas-phase mass fractions
<**COVERAGE**>	surface coverages (default: coverages from <SPECIES> section)

and the subsection **<INLET>**:

T=*d*	temperature [K]
<**MOLEFRAC**>	gas-phase mole fractions
<**MASSFRAC**>	gas-phase mass fractions
Vflux=*d*	inlet volume flux (at current T,p) [m^3/s]

9.3 Output

9.3.1 Screen output

On screen the integration progress is monitored. The values in the two columns displayed stand for time and temperature.

```
**************************************************************************
****                          DETCHEM_CSTR                            ****
****                   S. Tischer, O. Deutschmann                     ****
****                   VERSION 2.0.4 ** 05/05/31                      ****
**************************************************************************
    0.00E+00    1200.0
    5.63E-02    1098.5
    1.08E-01    1081.5
    ...
      100.00    1040.2
 CSTR finished successfully
```

9.3.2 *cstr***.plt*

DETCHEM^{CSTR} produces one output file ***cstr***.plt***. The output file is in TECPLOT format. TECPLOT will create a line graph upon loading the file ***cstr***.plt***. Nevertheless, this format also allows for easy import into spreadsheet programs like Microsoft Excel. The variables in the file are: time, temperature, species mole fractions (prefixed with X), and species mass fractions (prefixed with Y).

9.3.3 Examples

There is one example supplied along with the installation. The example directory contains the following files: ***cstr.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism ***noo2m***, ***thermdata***, ***moldata***, and a script file ***go***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

9.4 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.

9.5 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***batch***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/batch*** and call ***./batch***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 10

DETCHEM$^{\mathrm{STAG}}$

10.1 Introduction

The DETCHEM$^{\mathrm{STAG}}$ code simulates the behaviour of a catalytically active stagnation point flow reactor. Model equations couple flow equations together with the energy, species continuity and surface equations.

10.2 Governing Equations

In the stagnation point flow reactor, catalytically active surface interacts with the surrounding flow. Therefore, model equations consist of three parts, i.e., gas-phase equations, surface equations and boundary conditions.

10.2.1 Gas-phase equations

Temperature equation

$$\frac{\partial T}{\partial t} = -\left[\frac{\rho v_x}{\rho} + \frac{1}{\rho c_p}\sum_{i=1}^{N_g} c_{p,i}\, j_i\right]\frac{\partial T}{\partial x} - \frac{1}{\rho c_p}\sum_{i=1}^{N_g}\dot{\omega}_i M_i h_i + \frac{1}{\rho c_p}\frac{\partial}{\partial x}\left(\lambda\frac{\partial T}{\partial x}\right) \tag{10.1}$$

Species equation

$$\frac{\partial Y_i}{\partial t} = -\frac{\rho v_x}{\rho}\frac{\partial Y_i}{\partial x} + \frac{1}{\rho}\dot{\omega}_i M_i - \frac{1}{\rho}\frac{\partial j_i}{\partial x} \tag{10.2}$$

Continuity equation

$$0 = \frac{p}{R}\frac{\bar{M}^2}{T^2}\left[T\sum_{i=1}^{N_g}\frac{\partial_t Y_i}{M_i} + \frac{\partial_t T}{\bar{M}}\right] - 2\rho V - \frac{\partial(\rho v_x)}{\partial x} \tag{10.3}$$

Radial momentum equation

$$0 = -\frac{\rho v_x}{\rho}\frac{\partial V}{\partial x} - V^2 - \frac{\Lambda}{\rho} + \frac{1}{\rho}\frac{\partial}{\partial x}\left(\mu\frac{\partial V}{\partial x}\right) \tag{10.4}$$

The equation for the eigenvalue of the momentum

$$0 = \frac{\partial \Lambda}{\partial x} \tag{10.5}$$

Ideal gas law

$$\rho = \frac{p\bar{M}}{RT} \tag{10.6}$$

Axial velocity v_x appears in the equations. However ρv_x is considered to be an independent variable, not only the axial velocity. Heat capacity c_p, thermal conductivity λ and viscosity μ of the mixture, heat capacity $c_{p,i}$, molar mass M_i and enthalpy h_i of each species and gas-phase reaction rates $\dot{\omega}_i$ are calculated in the DETCHEM library. Diffusion velocity of each species is also introduced in the temperature and species equations. It is calculated at the mixture average level by the following equation.

$$\hat{j}_i = -\left(\rho D_{i,M}\frac{Y_i}{X_i}\frac{\partial X_i}{\partial x} + \frac{D_i^T}{T}\frac{\partial T}{\partial x}\right) \tag{10.7}$$

Diffusion velocities are corrected with the Eq. 10.8

$$j_i = \hat{j}_i - Y_i \sum_{k=1}^{N_g} \hat{j}_i \tag{10.8}$$

where Y_i is the mass fraction of each species. Molar mass of the mixture $\bar{M}$, averaged diffusion $D_{i,M}$ and thermal diffusion ${D_i}^T$ coefficient of each species are also calculated in the DETCHEM library. Please note that N_g represents the gas-phase species in the equations.

10.2.2 Surface equations

Chemical processes at the surface can be calculated based on three different approaches: Instantaneous diffusion at the gas/surface interface, detailed washcoat model and effectiveness factor approach. Instantaneous diffusion considers a flat surface where the catalyst is virtually distributed. Therefore, it does not account the internal mass transfer limitations. The detailed washcoat model and the effectiveness factor approach account the internal mass transfer limitations that are due to the porous washcoat layer. More detailed information can be found in section 2.3.

10.2.3 Boundary conditions

In order to completely describe the system, boundary conditions must also be included.

Gas-Surface/Washcoat Interface

In order to couple the catalytic surface and the surrounding flow, interaction between them must be accounted. Energy balance and species conservation equations are established at the interface. Energy balance is calculated as follows

$$\left(\rho c_p \Delta x^+\right) \frac{\partial T}{\partial t} = \lambda \frac{\partial T}{\partial x} - \sum_{i=1}^{N_g} \rho c_{p,i} Y_i \left(j_i + u\right) h_i - \sigma \varepsilon \left(T^4 - {T_{rad}}^4\right) - \sum_{i=1}^{N_g} \dot{s}_i M_i h_i + \dot{P} - \frac{\lambda_c}{d} \left(T - T_b\right) \tag{10.9}$$

Species governing equation accounts diffusion and convection processes. Due to the fact that species can be created or destroyed by the surface reactions, the source term is also included. The source term appears different based on the considered surface model. In case of instantaneous diffusion, the species governing equation at the gas/surface interface becomes

$$\rho \frac{\partial Y_i}{\partial t} \Delta x^+ = -j_i - \rho u Y_i + F_{cat/geo} \dot{s}_i M_i \tag{10.10}$$

Since there is mass flux to the surface, velocity cannot be zero. Therefore, Stefan velocity is considered in the species governing equation.

$$u = \frac{1}{\rho} \sum_{i=1}^{N_g} \dot{s}_i M_i \tag{10.11}$$

If the surface model is based on the detailed reaction-diffusion equations, the species governing equation at the gas-washcoat interface becomes

$$\rho \frac{\partial Y_i}{\partial t} \Delta x^+ = -j_i - \rho u Y_i + \dot{s}_i, eff M_i \tag{10.12}$$

where the diffusion flux at the gas/washcoat interface is treated as an effective surface reaction rate.

If the surface model is based on the effectiveness factor approach, the species governing equation at the gas-washcoat interface becomes

$$\rho \frac{\partial Y_i}{\partial t} \Delta x^+ = -j_i - \rho u Y_i + \eta F_{cat/geo} \dot{s}_i, eff M_i \tag{10.13}$$

Finally, continuity and radial velocity are also included at the interface.

$$\rho v_x := \rho u \qquad\qquad V := 0 \tag{10.14}$$

Inlet boundary

Inlet boundary conditions follow.

$$T = T^0 \qquad Y_i = Y_i^0 \tag{10.15}$$

$$\rho v_x = \rho^0 {v_x}^0 \qquad V = 0 \tag{10.16}$$

In order to close the equation system, a zero gradient condition is used for the mass flow at the inlet

$$\frac{\partial (\rho v_x)}{\partial x} = 0 \tag{10.17}$$

10.3 Solution

In DETCHEM^{STAG} the standard implicit code LIMEX is used to solve the coupled governing equations. PCHIPD package is used for the interpolations needed for the adaptive grid refinement procedure.

10.4 User Input

Before running DETCHEM^{STAG} the user must prepare an input file ***stag.inp***. An example is shown below.

```
{include species.inp}
{include mech.inp}

<SURFACE-MODEL>
  <CHEMSURF>
   time=10.
  </CHEMSURF>
    Fcatgeo = 1
</SURFACE-MODEL>

<STAGNATION>
  TIN    =  3.00E+02
  VIN    =     -0.50
  XLEN   =     0.030
  p      =     50000
  NGRID  =        60
  TIMES  =  0.00E+00
  TIMEF  =  1.00E+02
  HSTEP  =  1.00E-12
  ATOL   =  1.00E-06
  RTOL   =  1.00E-03
  ASPECT =  1.05
  TRANSIENTOUT=n
  TCONSTANT=n

  <MOLEFRAC>
      H2  5.00E-02
      AR  *
      O2  2.00E-02
  </MOLEFRAC>

  <SURFACE>
    TSURF   =  6.00E+02 [K]
    ALPHA   =  1.44E+00 [W/m/K]
    PINIT   =  2.50E+04 [W/m2]
    PLAST   =  3.00E+04
    PINCR   =  1.00E+03
    RFOI    =  9.00E-03
    TRAD    =  3.00E+02
    EPS     =  2.30E-01
    TBACK   =  3.00E+02
  </SURFACE>
</STAGNATION>
```

The file ***stag.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition, the files ***thermdata*** and ***moldata*** must be located in the executing directory.

Next, the **<SURFACE-MODEL>** tag must be included in the input file. Please see the surface models section for more detailed information.

The <STAGNATION> tag This is the opening tag for the stagnation input parameters. The following parameters must be included under this tag. If the default values are not given in the parenthesis for the parameters

they must be supplied by the user. Default values for some of the parameters can be used or they can also be supplied by the user.

```
TIN=d               Inlet gas temperature [K]
VIN=d               Axial inlet velocity [m/s]
XLEN=d              Surface-inlet distance [m]
p=d                 Pressure [Pa] (default:1.E5)
NGRID=i             Number of the grid points (default: 60)
TIMES=d             Initial integration time [s] (Default: 0.0)
TIMEF=d             Maximum integration time [s] (Default: 100)
HSTEP=d             Initial time integration step size [s] (Default: 1.E-12)
ATOL=d              Absolute tolerance [mol/m^3] (Default:1.E-6)
RTOL=d              Relative tolerance (Default:1.E-3)
ASPECT=d            Aspect ratio for the non-equidistant grid generation (Default: 1.0)
                    (An equidistant grid is generated if aspect=1.0)
THERMALDIFF=b       Thermal diffusion effect (Default: n)
                    (Thermal diffusion is important especially for light species, e.g. H2)
WASHCOAT=b          The effect of the washcoat is taken into account based on the detailed reaction-
                    diffusion equations. A direct coupling is considered between the washcoat and the
                    surrounding gas-phase (Default:n)
                    (A normal simulation is performed (without washcoat effect), if this option is not
                    invoked)
WASHCOAT2=b         The effect of the washcoat is taken into account based on the detailed reaction-
                    diffusion equations. An indirect coupling is considered between the washcoat and the
                    surrounding gas-phase. Indirect coupling implies that the washcoat is solved first.
                    After it reaches steady-state, it is coupled with the gas-phase at the interface. This
                    is repeated through the whole time integration of the system (Default:n)
                    (A normal simulation is performed (without washcoat effect), if this option is not
                    invoked)
WASHCOAT3=b         The effect of the washcoat is calculated based on the effectiveness factor approach
                    (Default:n)
                    (A normal simulation is performed (without washcoat effect), if this option is not
                    invoked)
TRANSIENTOUT=b      Enable transient solution output
                    (Default:n, in this case just the steady-state solution output will be given)
TCONSTANT=b         Activate the option to keep the surface temperature constant through whole solution
                    (Default:n)
```

The <MOLEFRAC> tag Mole fraction of the species entering from the inlet must be included within this tag.

The <SURFACE> tag The following surface parameters must/can also be supplied by the user within this tag.

```
TSURF=d             Surface temperature [K]
ALPHA=d             Thermal conductivity of the foil [W/m/K] (Default:1.44)
PINIT=d             (Initial) Power supplied to heat the catalytic surface [W/m^2] (Default: 0.0)
PLAST=d             (Last) Power supplied to heat the catalytic surface [W/m^2] (Default:0.0)
PINCR=d             (Step) Power supplied to heat the catalytic surface [W/m^2] (Default:0.0)
RFOI=d              Thickness of the foil [m] (Default: 0.003)
TRAD=d              Radiation temperature to the wall [K] (default: 300)
EPS=d               Emissivity of the surface (Default: 0.77)
TBACK=d             Backside temperature of the foil [K] (Default: 300)
```

As seen in the **<SURFACE>** tag, the user can give just one exact power input with the initial power *PINIT* parameter (in this case *PLAST* and *PINCR* are not required). It is also possible to give a linear power increase to the program by supplying *PINIT*, *PLAST*, *PINCR* values.

Estimated coverage of the surface species and surface site density must also be supplied to the code. An example is given below in the ***species.inp*** input file.

```
<SPECIES>
  <GASPHASE>
    AR
    H2
    H2O
    O
    OH
    H
  </GASPHASE>

  <SURFACE mol/cm2=  2.72E-09>
    PT(s)
    H2O(s)
    H(s)
    OH(s)
    <INITIAL>
      H(s)    4.0E-01
      PT(s)   *
      H2O(s)  2.0E-01
    </INITIAL>
  </SURFACE>
</SPECIES>
```

As seen on the above example, estimated coverage of the surface species are given within the **<INITIAL>** tag. Sum of the surface coverages must add upto 1. Surface site density is given with the unit of mol/cm^2 in the **<SURFACE>** tag declaration.

10.5 Output

DETCHEM^{STAG} creates five different output files and a screen output.

10.5.1 Screen output

A short information of the current solution will be given on the screen as given below.

```
**************************************************************************
****                        DETCHEM STAGNATION                        ****
****                           H.KARADENIZ                            ****
****                    VERSION 2.3.21 ** 2012/01/30                  ****
**************************************************************************
 ****   REGRID DONE    ****
Old grid points 60New grid points 64
 *****     REGRIDDING PROCESS COMPLETED     *****
 Calculation for:     0.00 W/m2  ,  Time:  0.00s
 Calculation for:     0.00 W/m2  ,  Time:  0.00s
 Calculation for:     0.00 W/m2  ,  Time:  0.01s
 Calculation for:     0.00 W/m2  ,  Time:  0.02s
 Calculation for:     0.00 W/m2  ,  Time:  0.04s
 Calculation for:     0.00 W/m2  ,  Time:  0.07s
      ...  ...      ...
      ...  ...           ...
      Limex finished successfully
```

DETCHEM^{STAG} has the ability to implement an adaptive refinement procedure during the grid generation (except the washcoat part). An information will be given about the implementation of the adaptive refinement if the user invokes the option. After the header, the next three lines will give a short information about the grid generation (whether it is done or it is not necessary). The number of the old and new grid points will be displayed. The next lines will give following: for which power the calculation is performed (for instance, 300.0 W/m2) and current integration time of the solution). The last line gives information whether the solution is finished successfully.

10.5.2 gas_***.dat

This file is generated according to the power input. For instance, if the user gives zero power input the output file will look like **gas_ 0.00dat**. If a power increase is supplied in the ***stag.inp***, the program will also give an output for each power increase (for instance **gas_ 300.00.dat**, **gas_ 350.00.dat**). The output files will consist of the following information: **TIME** (integration time of the solution), **R** (distance from the surface), **TEMP** (temperature at the spatial distances from the surface), **V** (scaled radial velocity at the spatial distances from the surface), **RHOVAU** (the variable ρv at the spatial distances from the surface, please note that 100 is added as a default value to the variable, the actual value is normally -100 of it), **LAMBDA** (eigenvalue of the momentum equation which will remain constant at the spatial distances from the surface) and **mole fractions** (mole fraction of each gas-phase species at the spatial distances from the surface).

10.5.3 surf_***.dat

This file is generated for the normal simulation (without washcoat effect) and the effectiveness factor approach. It is named according to the power input. For instance, if the user gives zero power input the output file will look like **surf_ 0.00dat**. If a power increase is supplied in the ***stag.inp***, the program will also give an output for each power increase (for instance **surf_ 300.00.dat**, **surf_ 350.00.dat**). The output files will consist of the following information: **TIME** (integration time of the solution), **TEMP** (temperature on the catalytic surface),**surface coverages** (coverages of the surface species)

10.5.4 wcoat_***.dat

This file is generated for the 'washcoat' and 'washcoat2' simulations. It is named according to the power input (as explained in the gas_***.dat file generation). The output files will consist of the following information: **TIME** (integration time of the solution), **R** (distance from the washcoat support side to the gas-washcoat interface; the interface is not included), **mole fractions** (mole fraction of each gas-phase species at the spatial distances)

10.5.5 wsurf_***.dat

This file is generated for the 'washcoat' and 'washcoat2' simulations. It is named according to the power input (as explained in the gas_***.dat file generation). The output files will consist of the following information: **TIME** (integration time of the solution), **R** (distance from the washcoat support side to the gas-washcoat interface; the interface is not included), **surface coverages** (coverages of the surface species at spatial distances)

10.5.6 wcstag_***.dat

This file is generated for the 'washcoat' simulation. It is named according to the power input (as explained in the gas_***.dat file generation). The output files will consist of the following information: **TIME** (integration time of the solution), **R** (distance from the washcoat support side to the inlet at the gas-phase; the gas-washcoat interface and the gas phase is consequently included), **mole fractions** (mole fraction of each gas-phase species at the spatial distances). The spatial distances are printed in 'mm' because of the low thickness of the washcoat. An illustration of the output follows.

```
TITLE = "STAG_WCOAT_AND_GASPHASE"
VARIABLES=
"                TIME"  "                   R"  "            CO      "  "            CO2     "
 0.100000000000000E+03   0.000000000000000E+00   0.486946019131708E-03   0.722022776070702E-01
 0.100000000000000E+03   0.219975998217398E-01   0.486946019131708E-03   0.722022776070702E-01
 0.100000000000000E+03   0.429476948900635E-01   0.486947118172136E-03   0.722022762284908E-01
 0.100000000000000E+03   0.629001663837050E-01   0.487295820737408E-03   0.722018319160494E-01
 0.100000000000000E+03   0.819025201871732E-01   0.587980625221570E-03   0.720735682409933E-01
 0.100000000000000E+03   0.100000000000000E+00   0.257785466728995E-01   0.399837172511616E-01
 0.100000000000000E+03   0.914846147153505E+01   0.558739720682844E-01   0.835443519329686E-03
 0.100000000000000E+03   0.186493460166468E+02   0.566910726668075E-01   0.676735726010959E-05
 0.100000000000000E+03   0.286252747890142E+02   0.566999270155862E-01   0.413898503389330E-07
 0.100000000000000E+03   0.391000000000000E+02   0.566999997202967E-01   0.118393880838135E-09
```

The washcoat thickness is taken 100μ m for the above example. 0 mm indicates there the washcoat support side, 0.1 mm indicates the gas-washcoat interface and 39.1 mm indicates the gas-inlet side.

10.6 Examples

There is one example supplied along with the installation. The example directory contains the following files: ***stag.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism ***gas_test***, a surface mechanism ***surf_test***,***mech.inp***, ***thermdata***, ***moldata*** and a script file ***go***.

10.7 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.

10.8 Running the tool

The example directory is supplied with a ***go*** script which can be used to run the tool just by typing ***go*** in the command prompt, or by any of the appropriate methods explained in the installation guide.

Chapter 11

DETCHEM$^{\mathrm{PLUG}}$

11.1 Introduction

The plug code simulates the behavior of plug flow chemical reactors. The code is designed for the non-dispersive one dimensional flow of chemically reacting ideal gas mixture. The following sections describe the DETCHEM$^{\mathrm{PLUG}}$ code that can be used to analyze the plug flow reactor for applications involving gases.

11.1.1 Governing Equations

For setting up the above equations it is assumed that (a) there is no variation in the transverse direction, and (b) axial diffusion of any quantity is negligible relative to the corresponding convective term.

$$\frac{d(\rho u A_c)}{dz} = A_s \sum_{k=1}^{kg} \dot{s_k}\, M_k \tag{11.1}$$

$$\rho u A_c \frac{dY_k}{dz} + Y_k A_s \sum_{k=1}^{kg} \dot{s_k}\, M_k = M_k (A_s\, \dot{s_k} + A_c\, \dot{\omega_k}) \tag{11.2}$$

$$\rho u A_c \frac{d(C_p T)}{dz} + \sum_{k=1}^{kg} \dot{\omega_k}\, h_k M_k A_c + \sum_{k=1}^{kg} \dot{s_k}\, h_k M_k A_s = U A_s (T_w - T) \tag{11.3}$$

$$PM = \rho R T \tag{11.4}$$

The above listed represents the total continuity, species continuity, energy, and the equation of state respectively. In-addition to these equation, since the residence time of gas is often a quantity of interest, a differential equation which computes it automatically is also included. This is simply

$$\frac{d\tau}{dz} = \frac{1}{u} \tag{11.5}$$

To improve the accuracy mass-transfer coefficients can be used additionally. They approximate the "resistance" to species mass transport between the mean composition and the composition at the reacting channel surface. The mass-transfer coefficients are defined by the following relationship

$$\dot{s_k}\, M_k = h_k (\rho_s Y_{k,s} - \rho_{k,m}) \tag{11.6}$$

where h_k is the mass-transfer coefficient, which can be represented in non-dimensional form using the species *Sherwood* numbers

$$Sh = \frac{h_k d}{D_{im}} \tag{11.7}$$

The *Sherwood* numbers are calculated locally using standard correlations. Since the heat and mass transfer phenomena in reacting flow is still an area of active research, the correlation used here represents the best efforts to-date.

11.1.2 Mass and Heat transfer coefficients

For the case of constant wall temperature the Nussult number is defined as

$$Nu_T = 3.657 + 8.827\left(\frac{1000}{Gz}\right)^{-0.545} exp\left(-\frac{48.2}{Gz}\right) \tag{11.8}$$

for the case of constant heat flux

$$Nu_H = 4.364 + 13.18\left(\frac{1000}{Gz}\right)^{-0.524} exp\left(-\frac{60.2}{Gz}\right) \tag{11.9}$$

Where Gz is the Graetz number for heat transfer. In the case of mass transfer the Sherwood numbers are expressed by the same equation with the Gz replaced by the Greatz number for mass transfer. Since the Greatz number is calculated as a function of reactor position, the mass and heat transfer coefficient calculated in DETCHEM^{PLUG}are the local values.

11.1.3 Solution

In DETCHEM^{PLUG} the standard implicit code LIMEX is used to solve the coupled governing equations. (Instead of LIMEX, DASPK3 can also be used, but for problems involving mass transfer coefficient, use of DASPK3 is not advised) DETCHEM^{PLUG} offers various options to solve the problem. The temperature solution offers **Isothermal**,**Non-isothermal, Adiabatic** and **Non-adiabatic** conditions. In addition to these, the axial temperature profile can be specified by a subroutine. In **Isothermal** as well as in the **Axial temperature** profile case the energy equation is not solved. The species transport can be modeled for **Gas-phase** and the **Surface** reactions. As a special case, the surface reactions can be specified to take place on **Washcoat**.
Though DETCHEM^{PLUG} is essentially a one dimensional model, it supports a number of geometrical configuration (The mass and heat transfer coefficients are specific to the flow geometry). The configurations supported currently are **cylindrical, cylindrical with gradually increasing cross-sectional area, triangular, square, hexagonal** and a **sinusoidal** geometry.

11.2 User Input

Before running DETCHEM^{PLUG}the user must prepare an input file ***plug.inp***. An example is shown below.

```
{include mech.inp}          # Mechanism input file
{include species.inp}       # Species input file
{include detchem.inp}       # optional
{include washcoat.inp}      # if washcoat reaction is desired
<PLUG>                      # opening tag for plug specific data
  <GEOMETRY factor=1 >      # opening tag for geometry specific data
                              factor stands for the type of geometry
   <CYLINDRICAl>            # opening tag for cylindrical geometry
     rin=0.025              # radius at reactor inlet
     rout=0.025             # radius at reactor outlet
   </CYLINDRICAL>           # closing tag for geometry
   <SINUSOIDAL>             # opening tag for sinusoidal geometry
     decay=n                # option to specify whether the geometry has
                              a decaying aplitude structure (y) or (n)
     decay-at=3.1415        # if decaying structure, then the starting
                              point for decay ( multiples of pi)
     amplitude=1.5          # amplitude of the sine wave
     step=0.001             # surface area integration factor for
                              sinusoidal geometry
     phase-shift=0.1        # desired phase shift
     decay-ratio=0.2        # rate at which decay should happen
   </SINUSOIDAL>            # closing tag for simusoidal geometry
   <OTHERS>                 # opeing tag for other supported geometries
     dim=1.d-03             # dimension for any geometry other then cylindrical
   </OTHERS>                # closing tag for other
   <SECTION>                # opening tag for cascading reactors
      length=1.0            # length of the zone
      Fcat/geom =1.         # ratio of catalyst to physical surface in this
                              zone
      tw=1200               # wall temperature of the reactor section
      washcoat=y            # washcoat reactions (y) or no (n)
      gasphase-reaction=y   # volumetric reactions (y) or (n)
      surface-reaction=n    # surface reactions (y) or (n)
   </SECTION>               # closing tag for cascade
  </GEOMETRY>               # closing tag for geometry
```

```
  <OP-CONDITIONS>          # opening tag to specify operating conditions
    isothermal=n           # isothermal condition (y) or no (n)
    adiabatic=n            # adiabatic condition  (y) or no (n)
    mass-transfer=y        # whether mass transfer resistance has to be enabled
    axial-temp-profile=n   # axial temperature profile supplied by
                             subroutine (y) else (n)
  </OP-CONDITIONS>         # closing tag for operating conditions
  <PROCESS>                # opening tag for process
  <PARAM>                  # opening tag for process parameters
    T-inlet=600            # inlet temperature of the gas mixture
    Ignite_at=1200         # external ignition temperature to initiate
                               the reaction
    Vel-inlet=0.1          # average velocity of the inlet gas mixture
    Pressure=1.d5          # pressure at inlet
    htceff                 # heat transfer coefficient
  </PARAM>                 # closing tag for process parameters
  <GASPHASE>               # opening tag to specify the gasphase species
  <MASSFRAC>or<MOLEFRAC>   # mass or mole fraction of the species
    NO2  0.1               # name and mass/mole fraction of 1'st gas species
    N2   *                 # name and mass/mole fraction of 2'nd gas species
  </MASSFRAC>or</MOLEFRAC> # closing tag for mass/mole frac
  <GASPHASE>               # closing tag for gasphase species
  </PROCESS>               # closing tag for species
  <SOLVER solver=limex> or   # opening tag for solver
  <SOLVER solver=daspk>
    step-size=0.d0         # minimum stepsize allowed for integration
    max-h=0.1              # maximum stepsize allowed for integration
    <TOLERANCE>            # opening tag for convergence tolerance
      <SPECIES>            # tag to specify tolerence for species equations
        absTol=1.d-03      # absolute tolerence for convergence
        relTol=1.d-06      # relative tolerence for convergence
      </SPECIES>           # closing tag for species tolerences
      <OTHERS>             # opeing tag for temperature tolerence
        absTol=1.d-03      # absolute tolerence for temperature equation
        relTol=1.d-05      # relative tolerence for temperature equation
      </OTHERS>            # closing tag for temperature tolerances
    <TOLERANCE>            # closing tag for tolerences
  </SOLVER>                # closing tag for solver
  <CHEMSURF>               # opening tag for chemsurf options
    hini=1.d-10            # inital time-step for integration
    time=100               # end time for integration for surface coverages
  </CHEMSURF>              # closing tag for chemsurf
  <OUTPUT>                 # opening tag for output options
    file=6                 # integer number to be appended at the end
                             of output file names
    title=noo2             # required title in the output file
    transfer-coeff=y       # transfer coefficient outputs (y) or (n)
    monitor=1              # option to monitor the solution progress
    coverage=y             # coverage output (y) or (n)
    species=y              # mass or mole output option for the
    species
  <SUMMARY>
    sp = NO                # name of species whose exit conditions required
    sp = O2                # name of species whose exit conditions required
  </SUMMARY>
  <SUM>
    sp = NO2
    sp = N2
  </SUM>
  </OUTPUT>
</PFR>
```

The file ***plug.inp*** holds all the parameters (which are grouped into different sections) required to setup the problem. The file must start with the include statements for ***species.inp***,***mech.inp***, and ***washcoat.inp*** (if washcoat reaction is desired). Alternatively the pre-processed input file ***detchem.inp*** can be included.
The following table specifies the different geometries that can be used with mass and heat transfer coefficients.

```
Factor   Shape
1        regular cylinder
2        circular cylinder with steadily varying c/s area
3        square
4        equilateral triangle
5        hexagonal
6        sinusoidal
```

In the input file when factor is given the value 1, then plug assumes a regular cylinder and *rout* will be set equal to *rin*. The rout option in the cylindrical tab is effective only when factor is given the value 2. For any value of factor greater than 2, the dimension given in the OTHERS tab will be used to calculate the surface area and cross sectional area.

If *htceff* is supplied in process parameters, then the user supplied value will be used while solving the energy equation. On the other hand the user can comment the tag *htceff* then the heat transfer coefficient will be calculated using correlations.

If axial temperature profile is opted in operating conditions then a subroutine named *PFR_- TempProfile_Usr* must be supplied. A pseudo code for subroutine TempProfile_Usr is supplied along with the installation. Note that the point temperatures must be stored in the variable *tp*. z is the position along the length of the reactor. The user can modify the code and recompile the files using Makefile.

The convergence rate can vary from case to case. In any case if the code fails during the run, then the user is advised to change the tolerance parameters. There are no strict guidelines for setting the tolerance.

11.2.1 The <GEOMETRY> tag

The important point to be noticed in the **<GEOMETRY>** tag is the option **factor**. The values assigned for the factor decides the reactor shape under consideration. The values that can Ga assigned to **factor** is explained above.

11.2.2 The <SECTION> tag

By this feature the reactor can be divided into a number of sections and each section can be assigned with different process/boundary conditions. A total of 20 sections are supported. If there is only one **<SECTION>** tag, then the reactor is assumed to be of single section with the length of the reactor equal to the length of the section. If more than one **<SECTION>** is specified, then the length of reactor is equal to the total length of the sections. And the simulation of each reactor section will be carried out according to the conditions specified under each **<SECTION>** tag.

11.2.3 The <SOLVER> tag

This tag holds the option for choosing the solver and the tolerance levels that can be assigned for different solution variables. In any case if the solver is unable to finish the calculation, it is advisable to relax the tolerance criteria as well as the **maxh**, especially when calculations are performed with mass transfer option.

11.2.4 The <SUMMARY> and <SUM> tag

In the plug input file, there is an option to produce summary of the simulation, which is nothing but the exit conditions of the chemical species, In case if there are large number of species involved in the problem, and the user wants to know the exit conditions of certain specific species, then the name of those species can be listed under the **<SUMMARY>** tag and hence the exit conditions will be written in the file ***summary.dat*** .

If the user wants to find the sum of mass/mole fraction of certain species up on exit, then the **<SUM>** tag can be used, which will produce the sum of the mass/mole fraction of the species in the output file ***summary.dat***.

11.3 Output

Depending on the options set in the input file DETCHEM^{PLUG} produces several file outputs. All the output files are in Tecplot format.

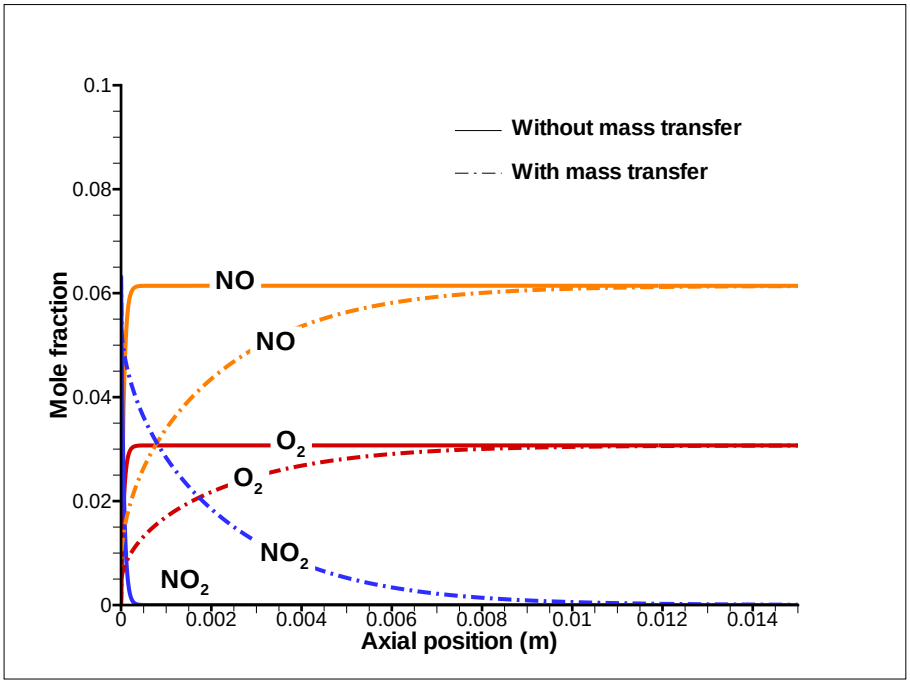
11.3.1 Screen output

Other than the file output, DETCHEM^{PLUG} produces screen output depending on the monitor option specified in the input file. With *monitor=1*, the screen output reports the iteration count and the temperature. With *monitor=2*, it reports the iteration count, temperature and the species mass fractions. An example of screen output is shown below.

```
**************************************************************************
****                          DETCHEM PLUG                            ****
****    Vinod M Janardhanan, Steffen Tischer & Olaf Deutschmann       ****
****                    VERSION 2.0 ** 01,01,2004                     ****
****                      ---------------                             ****
**************************************************************************
 Iterations
       1    0.00000  700.00000
       2    0.00000  698.27944
       3    0.00000  700.85369
 .    .......  .........
 .    .......  .........
      56    0.04867 1200.01246
      57    0.04967 1200.01246
      58    0.05000 1200.01199
 Elapsed Execution time  :   0.180971995
 User time  :   0.175972998
 System time  :   0.00499900011
```

11.3.2 *outg***.plt*

The output file ***outg***.plt*** lists velocity, residence times, temperature density and the gas-species mass or mole fractions, depending on whether the species tag in **<OUTPUT>** element is supplied with *mass* or *mole* option.



11.3.3 *outs***.plt*

If *coverage* option is chosen in then the file ***outs***.plt*** will be produced,which lists the surface coverages along the axial position of the reactor.

11.3.4 *trans***.plt*

If *transfer-coeff* option is chosen then the file ***trans***.plt*** will be produced which will list the mass and heat transfer coefficient calculated using the correlations.

11.3.5 *summary.dat*

The file ***summary.dat*** contains the exit temperature and composition of the species listed in the **<SUMMARY>** tag. If the **<SUM>** tag is also used then the file ***summary.dat*** also contains the sum of the mass/mole fraction of the species listed in the **<SUM>** tag.

11.4 Examples

There are a few examples supplied along with the installation. Each example directory contains the following files: ***plug.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism file, a surface mechanism file, ***thermdata***, and ***moldata***. The optional washcoat input has been saved in a separate file ***washcoat.inp***.

11.5 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.In case of problems, adjusting the solver and tolerance parameters may improve the performance.

11.6 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***./plug***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/plug*** and call ***./plug***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 12

DETCHEM$^{\text{PACKEDBED}}$

12.1 Introduction

Since the discovery of solid catalysts, packed bed reactors become an important class of reactors in chemical industry. They are the most commonly fund reactors in Petroleum and Petrochemical industry. There are a number of ways by which one can represent a packed bed reactor mathematically and hence to solve it numerically. The most commonly found models are 1) One-dimensional homogeneous 2) two- dimensional homogeneous model 3) One-dimensional heterogeneous model 4) two-dimensional heterogeneous model. DETCHEM$^{\text{PACKEDBED}}$ is basically a one-dimensional heterogeneous model. The one-dimensional heterogeneous model assumes that there is no radial variations in the flow properties. Most of the industrial reactors runs under adiabatic conditions. The model presented here can be used for any operating conditions. The model do not takes care of axial diffusion.

12.1.1 Governing Equations

For setting up the above equations it is assumed that (a) there is no variation in the transverse direction, and (b) axial diffusion of any quantity is negligible relative to the corresponding convective term.

$$\frac{d(\rho u)}{dz} = a_v \sum_{k=1}^{kg} \dot{s_k}\, M_k \tag{12.1}$$

$$\rho u \frac{dY_k}{dz} + Y_k a_v \sum_{k=1}^{kg} \dot{s_k}\, M_k = M_k (a_v\, \dot{s_k} + \dot{\omega_k}\, \epsilon) \tag{12.2}$$

$$\rho u \frac{d(C_p T)}{dz} + \sum_{k=1}^{kg} \dot{\omega_k}\, h_k M_k \epsilon + \sum_{k=1}^{kg} \dot{s_k}\, h_k M_k a_v = \frac{4}{d_h} U A_s (T_w - T_f) \tag{12.3}$$

$$\frac{dp}{dz} = f \frac{(\rho u^2)}{d_p} \tag{12.4}$$

$$PM = \rho RT \tag{12.5}$$

The above listed represent the total continuity, species continuity, energy, and the equation of state respectively. In-addition to these equations, since the residence time of gas is often a quantity of interest, a differential equation which computes it automatically is also included. This is simply

$$\frac{d\tau}{dz} = \frac{1}{u} \tag{12.6}$$

To improve the accuracy mass-transfer coefficients are used to approximate the "resistance" to species mass transport between the mean composition and the composition at the reacting catalyst surface. The mass-transfer coefficients are defined by the following relationship

$$\dot{s_k}\, M_k = h_k (\rho_s Y_{k,s} - \rho_{k,m}) \tag{12.7}$$

The mass transfer coefficients can be represented in non-dimensional form using the species *Sherwood* numbers

$$Sh = \frac{h_k d}{D_{im}} \tag{12.8}$$

The *Sherwood* numbers are calculated using standard correlations.. Since the heat and mass transfer phenomena in reacting flow is still an area of active research, the correlation used here represents the best efforts to-date. Inter-phase heat transfer between the solid and the fluid phase inside the reactor leads to difference in temperature between the solid and fluid phase. Therefore the surface reaction rate has to be evaluated at the prevailing surface temperature of the catalyst. The heat transfer inside the reactor reads as follows.

$$h_{fs}(T_s - T_f) = \sum_{k=1}^{kg} \dot{s_k}\, h_k M_k \tag{12.9}$$

12.2 Solution

In DETCHEM^{PACKEDBED} the standard implicit code LIMEX is used to solve the coupled governing equations. DETCHEM^{PACKEDBED} offers various options to solve the problem. The temperature solution offers **Isothermal**, **Non-isothermal**, **Adiabatic** and **Non-adiabatic** conditions. The species transport can be modeled for **Gas-phase** and the **Surface** reactions

12.3 User Input

Before running DETCHEM^{PACKEDBED} the user must prepare an input file ***pack.inp***. An example is shown below

```
{include species.inp}      # Species input file
{include mech.inp}         # Mechanism input file
<PACK>                     # opening tag for plug specific data
  <GEOMETRY model=channel> # opening tag for geometry specific data
     rin=0.025             # inner radii of the reactor
     <SECTION>
      length= 0.05         # length of the reactor section
      gaschem = y          # gas-phase chemistry (y) or (n)
      surfchem= y          # surface reactiosn (y) or (n)
      t-wall=1000          # wall temperature of the reactor section
      porosity= 0.37       # void fraction of the packing
      particle_dia=3.7d-03 # equivalent diameter of the packing particles
      kp=0.3               # thermal conductivity of the solid packings
      area/volume=400      # porous area to volume ratio
     </SECTION>
     <WALL>
      thickness            # thickness of the reactor wall
      conductivity         # thermal conductivity of the reactor wall
      ho                   # outside heat transfer coefficient
     </WALL>
  </GEOMETRY>              # closing tag for geometry
  <INLET>
    velocity=1             # inlet velocity of the fluid stream
    temperature=300        # inlet temperature of the fluid stream
    pressure=1d05          # inlet pressure of the fluid stream
    <GASPHASE>
     <MASSFRAC>/<MOLEFRAC> # mass  or mole fraction of the species
       NO2  0.             # name and mass/mole fraction of 1'st gas species
       N2   *              # name and mass/mole fraction of 2'nd gas species
     </MASSFRAC>/</MOLEFRAC>
    </GASPHASE>
  </INLET>
  <PROCESS>
    isothermal= y          # isothermal conditions (y) or (n)
    adiabatic=n            # adiabatic conditions (y) or (n)
    mass-transfer=y        # mass-transfer resistance (y) or (n)
  </PROCESS>
  <CHEMSURF>               # opening tag for chemsurf options
    hini=1.d-10            # inital time-step for integration
    time=100               # end time for integration for surface coverages
  </CHEMSURF>              # closing tag for chemsurf
  <SOLVER solver = limex>  # opening tag for solver using limex solver
    hini = 0.d0            # initial step size for integration
    hmax =0.1              # maximum stepsize allowed for integration
    <TOLERANCE>            # opening tag for convergence tolerance
      <MOMENTUM>           # tag to specify tolerence for momentum equation
        aTol= 1.d-06       # absolute tolerence for convergence
        rTol= 1.d-06       # relative tolerence for convergence
      </MOMENTUM>          # closing tag for momentum tolerence
      <SPECIES>            # tag to specify tolerence for species equations
        aTol=1.d-06        # absolute tolerence for convergence
```

```
        rTol=1.d-06        # relative tolerence for convergence
      </SPECIES>           # closing tag for species tolerences
      <TEMPERATURE>        # opeing tag for temperature tolerence
        aTol=1.d-06        # absolute tolerence for temperature equation
        rTol=1.d-06        # relative tolerence for temperature equation
      </TEMPERATURE>       # closing tag for temperature tolerences
    <TOLERANCE>            # closing tag for tolerances
  </SOLVER>                # closing tag for solver
  <OUTPUT>                 # opening tag for output options
    file=6                 # integer number to be appended at the end
                             of output file names
    title=noo2             # required title in the output file
    monitor=1              # option to monitor the solution progress
    coverage=y             # coverage output (y) or (n)
    species=mass           # mass or mole output option for the species
  </OUTPUT>
</PFR>
```

12.3.1 The <SECTION> tag

With this tag the total reactor length can be divided into a maximum of 20 sections. And each section can be assigned different properties. For eg. the porous media properties can be assigned different values as well as a particular part of the porous media can be assumed to be inert and so on.

12.3.2 The <WALL> tag

When it is desired to take care of the heat transfer from the outside wall, this tag must be used and the wall thickness must be given. In case if the thickness is assigned a value 0 then the conduction within the wall will not be taken care in the simulation.

12.3.3 The <SOLVER> tag

Though the solver option under the solver tag lets the user to switch between LIMEX and DASSL, it is highly recommended to use LIMEX in this release of the code.

12.3.4 The <TOLERANCE> tag

The convergence rate can vary from case to case. In any case if the code fails during the run, then the user is advised to change the tolerance parameters as well as *hmax* under the **<SOLVER>** tag. i.e lowering *hmax* and increasing tolerances can help in convergence. There are no strict guidelines for setting the tolerance.

12.3.5 The <OUTPUT> tag

The **<OUTPUT>** tag contains many options for writing the output file. Depending on the options set in the **<OUTPUT>** tag, different variables will be writen in the output files.

12.4 Output

Depending on the options set in the input file ***pack.inp*** DETCHEM$^{\text{PACKEDBED}}$ produces several output files. The most important ones being the ***outg***.plt*** and ***outs***.plt***.

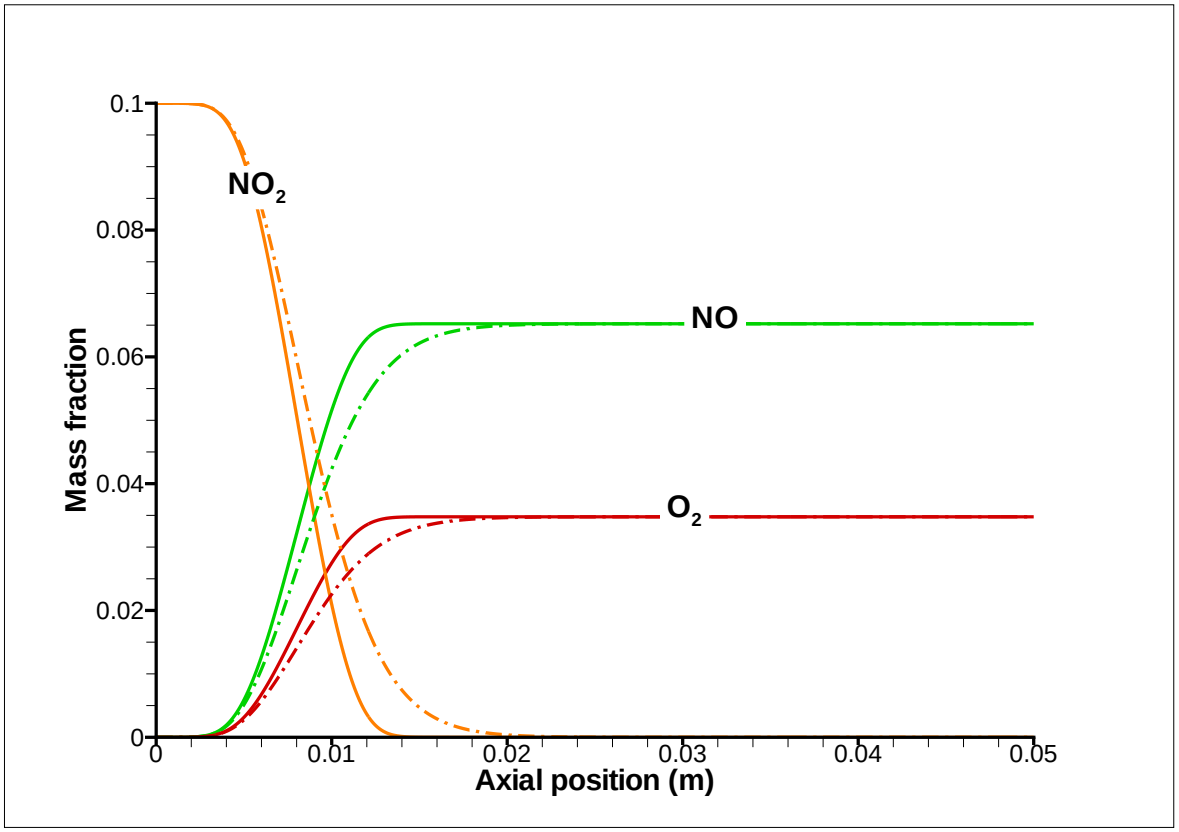
12.4.1 Screen output

Other than the file output, DETCHEM$^{\text{PACKEDBED}}$ produces screen output depending on the monitor option specified in the input file. With *monitor=1*, the screen output reports the iteration count and the temperature. With *monitor=2*, it reports the iteration count, temperature and the species mass fractions. An example is shown below.

```
************************************************************************
****                      DETCHEM PACKEDBED                         ****
****    Vinod M Janardhanan, Steffen Tischer & Olaf Deutschmann     ****
****                    VERSION 2.0 ** 01/01/2004                   ****
****                      ----------------                          ****
************************************************************************
        1    0.00000  500.00000
        2    0.00010  506.83525
  .    .......  .........
  .    .......  .........
      501    0.05000 1197.60715
 Elapsed Execution time :   1.71973801
 User time :   1.71373904
 Systemtime :   0.00599899981
```

12.4.2 *outg***.plt*

The ***outg***.plt*** contains mass/mole fraction of the gas-phase chemical species depending on the option given in the **<OUTPUT>** tag in the input file ***pack.inp***. The ***outg***.plt*** also contains the solid and fluid temperature designated as Tf and Ts respectively.



12.4.3 *outs***.plt*

outs.plt*** contains the surface coverages for the surface adsorbed chemical species and the fluid ans solid temperatures.

12.5 Examples

There are a few examples supplied along with the installation. Each example directory contains the following files: ***plug.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism file, a surface mechanism file, ***thermdata***, and ***moldata***.

12.6 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.In case of problems, adjusting the solver and tolerance parameters may improve the performance.

12.7 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***pbed***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/pbed*** and call ***./pbed***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 13

DETCHEM$^{\text{CHANNEL}}$

13.1 Introduction

The DETCHEM$^{\text{CHANNEL}}$ code simulates the steady state chemically reacting gas flow through a cylindrical channel using the boundary layer approximation. Detailed mechanisms for surface and gas phase reactions are considered.

13.1.1 The boundary layer approach

In the most general case of simulating the flow field of a fluid, the governing equations are of Navier-Stokes type. However, due to their mathematical structure, finding a solution can become very time and resource consuming. In order to get a faster algorithm, one possible simplification is given by the boundary layer approximation. It is suitable for systems with a main direction of the convective flow, in which the diffusive transport along this direction is negligible compared to the convection. This assumption becomes valid for a channel or any cylindrical reactor with sufficiently high velocity of the fluid or sufficiently small diameter of the channel. All other transport effects within the fluid, e.g. diffusion limitations of surface reaction rates, are considered in this approximation.

13.1.2 Governing Equations

$$\frac{\partial(r\rho u)}{\partial z} + \frac{\partial(r\rho v)}{\partial r} = 0 \tag{13.1}$$

$$\frac{\partial(r\rho u^2)}{\partial z} + \frac{\partial(r\rho uv)}{\partial r} = -r\frac{\partial p}{\partial z} + \frac{\partial}{\partial r}\left(\mu r\frac{\partial u}{\partial r}\right) \tag{13.2}$$

$$\frac{\partial p}{\partial r} = 0 \tag{13.3}$$

$$\frac{\partial(r\rho uh)}{\partial z} + \frac{\partial(r\rho vh)}{\partial r} = u\frac{\partial p}{\partial z} + \frac{\partial}{\partial r}\left(\lambda r\frac{\partial T}{\partial r}\right) - \frac{\partial}{\partial r}\left(\sum_s r j_s h_s\right) \tag{13.4}$$

$$\frac{\partial(r\rho uY_s)}{\partial z} + \frac{\partial(r\rho vY_s)}{\partial r} = -\frac{\partial}{\partial r}(r j_s) + r\dot{\omega}_s \tag{13.5}$$

The above listed represent the total continuity, axial momentum, radial momentum, the energy and the species continuity respectively.

The flux j_s is calculated by

$$j_s = \begin{cases} \dot{r_s} & \text{if } r = r_{\min} \\ -\rho D_s \frac{M_s}{M}\frac{\partial X_s}{\partial r} & \text{if } r_{\min} < r < r_{\max} \\ -\dot{r_s} & \text{if } r = r_{\max} \end{cases} \tag{13.6}$$

where
z - axial coordinate, r - radial coordinate, u - axial component of velocity, v - radial component of velocity, p - pressure, h - enthalpy density, Y_s - mass fraction of species s, X_s - mole fraction of species s, μ - viscosity, λ -

thermal conductivity, h_s - species enthalpy, $\dot{\omega}_s$ - gas-phase reaction rate, and j_s - the radial diffusion flux.

13.1.3 Solution

After the radial discretization of the above equations into n cells with boundary coordinates $r_0 \dots r_n$ one gets a system of first order differential equations in terms of z coordinate. The aim is to solve the system by method of lines. However, the mathematical structure of equation system is not suitable for the used stiff-stable integrator LIMEX. Therefore some transformation of the system is done.

A finite volume method is used to solve the system. The dependent variables are defined as (the index i denotes the number of the cell):

mass flux

$$\dot{m}_i = 2\pi \int_{r_{i-1}}^{r_i} r' \rho u dr' \tag{13.7}$$

flux of axial momentum

$$\dot{p}_i = 2\pi \int_{r_{i-1}}^{r_i} r' \rho u^2 dr' \tag{13.8}$$

enthalpy flux

$$\dot{H}_i = 2\pi \int_{r_{i-1}}^{r_i} r' \rho u h dr' \tag{13.9}$$

and species mass flux

$$\dot{m}_{s,i} = 2\pi \int_{r_{i-1}}^{r_i} r' \rho u Y_s dr' \tag{13.10}$$

The system of equations are then transformed by using the mass flux instead of radial coordinate as an independent variable. This will satisfy equation (1) and the terms containing the radial velocity v vanish. Here, the boundary layer equations are solved as follows:

- axial momentum

$$\frac{\partial \dot{p}_i}{\partial z} = \left[-\pi r^2 p_z + 2\pi r \mu \frac{\partial u}{\partial r} \right]_{r_{i-1}}^{r_i} \tag{13.11}$$

- energy

$$\frac{\partial \dot{H}_i}{\partial z} = \left[\pi r^2 u p_z + 2\pi r \lambda \frac{\partial T}{\partial r} \right]_{r_{i-1}}^{r_i} \tag{13.12}$$

- species mass

$$\frac{\partial \dot{m}_{s,i}}{\partial z} = \left[\pi r^2 \dot{\omega}_s - 2\pi r j_s \right]_{r_{i-1}}^{r_i} \tag{13.13}$$

where $p_z = \frac{\partial p}{\partial z}$ is used as abbreviation.

The radial coordinate as a dependent variable is calculated by an algebraic equation derived from equation(7). The temperature T is determined implicitly through the dependence of the enthalpy on temperature and species composition. The coefficients μ, λ, D_s are also functions with both dependencies.

The pressure gradient p_z must be determined consistently in order to satisfy the boundary conditions that the radius of the outermost cell is fixed to $r_{\max}$. For reasons of numerical stability of the solver LIMEX, a differential equation is used, which provides a relaxation mechanism to find the correct pressure gradient.

$$\alpha_p \frac{\partial p_z}{\partial z} = p_{z,\mathrm{ref}} \left(1 - \frac{r_n}{r_{\max}} \right) \tag{13.14}$$

The factor α_p can be thought of as the relaxation length of the pressure gradient.

The coupled solution of governing equations in DETCHEM$^{\mathrm{CHANNEL}}$ is attained by the standard implicit code LIMEX.

13.2 User Input

Before running DETCHEM$^{\text{CHANNEL}}$ the user must prepare an input file ***channel.inp***. An example is shown below.

```
{include species.inp}
{include mech.inp}

 <SURFACE-MODEL>
   <CHEMSURF>
     hini=1.d-10
     time=1.d0
   </CHEMSURF>
   <Fcatgeo>
     Pt   1
   </Fcatgeo>
   {include washcoat.inp}
 </SURFACE-MODEL>

 <CHANNEL Version=2.0>
 <BASICS>
   title="EXAMPLE DETCHEM V2.0"  # title used in the output file
   ngrid=15                      # number of grids in the radial direction
   zmax=0.1                      # length of the channel [m]
   rmax=0.001                    # radius of the channel [m]
 </BASICS>

 <SECTION>
   gaschem=y                     # gas-phase chemistry (yes/no)
   <WALL>
     surfchem=y                  # surface chemistry (yes/no)
     mechanism="Pt"              # name of the surface mechanism
   </WALL>
 </SECTION>

 <SOLVER>
   hini=1.d-10                   # initial step size [m]
   hmax=0.01                     # maximum step size [m]
 </SOLVER>

 <OUTPUT>
   outg=y                        # gas-phase output file (y/n)
   outs=y                        # surface output file (y/n)
   summary=y                     # add a line to summary_out.dat (y/n)
   file=1                        # file no. appended to output file names
   copy=y                        # save a copy of this input file (y/n)
   monitor=1                     # screen output (0=none, 1=normal, 2=extended)
 </OUTPUT>

 <INLET>
   T0=300                        # inlet temperature [K]
   u0=0.1                        # inlet velocity [m/s]
   p=1.00e5                      # static pressure [Pa]
   <MOLEFRAC>
     NO2  0.1                    # inlet mole fraction
     N2   *                      # balance mole fraction
   </MOLEFRAC>
 </INLET>

 <TPROFILE>
   <DISCRETE>
     0.000    1123               # z-position[m], temperature [K]
   </DISCRETE>
 </TPROFILE>

 </CHANNEL>
```

The file ***channel.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition the files ***thermdata*** and ***moldata*** must be located in the executing directory. Alternatively, the pre-processed input file ***detchem.inp*** can be included.

Next, an optional washcoat definition may appear in the input file. If washcoat models shall be applied, the user must supply this information. However, by including the washcoat definition the model is not activated automatically. Use the **washcoat** member in the **<SECTION>** tag to activate the model.

The options within the **<CHANNEL>** tag are as follows.

Version=*s*	input file version (not evaluated)
name=*s*	channel name (used in multi-channel simulations) (default: none)
<**BASICS**>	basic options (must appear as first tag)
<**SECTION**>	channel properties (channel may be devided into sections)
<**SOLVER**>	DAE solver parameters
<**TOLERANCE**>	tolerance parameters
<**OUTPUT**>	output options
<**INLET**>	inlet specifications
<**TPROFILE**>	wall temperature profile

The optional member **name** is only required for setting up multi-channel simulation with the tool DETCHEM^{MONOLITH}.

13.2.1 The <BASICS> tag

The options within the **<BASICS>** tag are:

title=*s*	title (used in TECPLOT output) (default: 'DETCHEM-CHANNEL')
ngrid=*i*	number of radial grid points
zmin=*d*	starting position of the channel (default: 0)
zmax=*d*	end point of the channel (or length)
rmin=*d*	inner radius of an annular channel (default: 0)
rmax=*d*	radius of the channel

DETCHEM^{CHANNEL} was desinged to simulate cylindrical channels. Nevertheless, the same approach is also valid for annular reactors, i. e. cylinder geometries with an inner and an outer wall. Use the **rmin** member to specify the radius of an inner wall.

13.2.2 The <SECTION> tag

The user can divide the channel in to several (default: up to 5) sections. This is desired when you have different catalyst properties along the length of the channel, or when the wall surface reactions are happening only at certain specific part of the reactor.

In each of the sections, you can specify the following options:

to=*d*	end point of section (default: zmax)
gaschem=*b*	enable gas-phase chemistry (default: no)
<**INWALL**>	inner wall boundary conditions for an annular channel (default: none)
<**WALL**>	wall boundary conditions

whereas the **<WALL>** and **<INWALL>** subsections contain

boundary=*b*	enable flow boundary condition (default: yes)
surfchem=*b*	enable catalytic surface reactions (default: no)
adiabatic=*b*	adiabatic wall conditions (default: no)
Fcatgeo=*d*	ratio catalytic / geometric surface area (default: 1)
mechanism=*s*	surface mechanism name (default: first surface mechanism defined in <MECHANISM>)

The member **Fcatgeo** defines an additional surface factor for this section. However, this factor cannot distinguish between different surface types. So it applies to all surface reactions in the same way. If the user wants individual catalytic surface areas, declare them in the **<SURFACE-MODEL>** section. If $F_{\mathrm{cat/geo}}$ is defined in both **<SECTION>** and **<SURFACE-MODEL>**, then both apply to the calculation of the reaction rates, i. e. the rates are multiplied by the product of the two values.

WARNING !!!: Declaration of **Fcatgeo** in this section may give wrong results when CHANNEL is used as part of other applications involving the calculation of transient coverages. This value is not passed to higher-order applications!

13.2.3 The <SOLVER> tag

In this section the user can define solver specific options:

hini=*d*	initial integration step size [m] (default: 1e-10)
hmin=*d*	minimum step size [m] (default: 1e-20)
hmax=*d*	maximum step size [m] (default: zmax)
chemmax=*i*	maximum number of iterations with chemical source terms (default: unlimited)
p_relax=*d*	pressure equation relaxation length [m] (default: 1e-10)
bulk=*s*	bulk species, i.e. species for that no conservation equation is solved (default: none)

The members **hini**, **hmin**, and **hmax** define the step size of the integration process. If the integration process fails, lowering these values may help. **p_relax** defines the relaxation length in the pressure correction equation. It should be as small as possible. However by decreasing this parameter, the differential equation system becomes more stiff.

The parameter **chemmax** defines a maximum number of integration steps, for that the chemical source terms are considered. This parameter is only useful in applications where DETCHEM$^{\text{CHANNEL}}$ is called automatically multiple times, e. g. in DETCHEM$^{\text{MONOLITH}}$. By limiting the number of iterations, the user can avoid channel simulations that need too much time to be solved. Although the result is incorrect and an error flag is passed, the simulation may continue.

With the **bulk** member, the user can specify a diluting species for that no transport equation will be solved. The mass fraction of this species always balances to unity. It is recommended that this species is always present (e.ġ. nitrogen or argon) and that this species is the last species in the gas-phase list.

13.2.4 The <TOLERANCE> tag

In this optional section the user may define tolerances for the system variables:

<**MDOT**>	mass flux [kg/s]
<**PDOT**>	momentum flux [kg m/sˆ2]
<**HDOT**>	enthalpy flux [J/s]
<**RADIUS**>	radial grid [m]
<**TEMPERATURE**>	temperature [K]

Each of the subtags may contain the following to parameters:

atol=*d*	absoulte tolerance
rtol=*d*	relative tolerance

In most cases, the default values work fine. If the simulation is not successful, changing these parameters may help.

13.2.5 The <OUTPUT> tag

The options in this section define the screen and file output of DETCHEM$^{\text{CHANNEL}}$

outg=*b*	enable gas-phase output file, e.g. outg001.plt (default: no)
outs=*b*	enable surface output file, e.g. outs001.plt (default: no)
outflux=*b*	enable output file for axial species fluxes (default: no)
summary=*b*	enable output in file summary_out.dat (default: no)
file=*i*	file number used as file name extension (default: 1)
copy=*b*	save copy of input file, e.g. channel001.inp (default: no)
minstep=*d*	minimum step size for writing output [m] (default: none)
monitor=*i*	monitoring options : 0= no monotoring, 1= monitor integration progress, 2= additional screen output (default: 0)
molefrac=*b*	write mole fractions instead of mass fractions (default: no)

The output files will be described later in this chapter. Each of the output file filenames (except ***summary_out.dat***) contains a three-digit extention with the filenumber specified by the **file** member. For parameter studies it may be useful to log the input for a specific cas. If the **copy** option is activated, a copy of the input file ***channel.inp*** will be saved with the same file extention as the other output files.

The **monitor** option determines the screen output during the integration process. There are three levels of output: none (0), normal (1), extended (2). In a single run of DETCHEM$^{\text{CHANNEL}}$ it is useful to monitor the integration process. By doing this, the user may get useful information about the speed of calculation and its convergence. In case of unsuccessful simulations, the extended output can write additional information. For multiple channel calculations, e. g. using DETCHEM$^{\text{MONOLITH}}$, we recommend to switch off monitoring of the channel calculations.

13.2.6 The <INLET> tag

In this section the user specifies the gas-phase properties at the inlet. Following options can be specified:

T0=*d*	inlet temperature [K]
u0=*d*	inlet velocity [m/s]
p=*d*	pressure [Pa] (default: 101325)
p/bar=*d*	pressure [bar]
<**MOLEFRAC**>	inlet mole fractions
<**MASSFRAC**>	inlet mass fractions

Note that the velocity **u0** is the actual inlet velocity at the given temperature **T0** and pressure **p**. It is up to the user to convert volume fluxes (e. g. in slpm) in to velocities (in m/s)

13.2.7 The <TPROFILE> tag

Specify the wall temperature in this section. The following options apply:

ignite_at=*d*	ignition temperature [K] – acts as a minimum temperature in adiabatic simulation (default: 0)
ignore_below=*d*	minimum temperature [K] for considering chemical reactions (default: 0)
adiabatic=*b*	use adiabatic wall (yes) or temperature profile (no) (default: no)
<**DISCRETE**>	piecewise constant temperature profile
<**LINEAR**>	piecewise linear temperature profile
<**ISOTHERM**>	multiple simulation runs with isothermal conditions

In order to define a temperature profile along the channel wall for a single run, use the tags **<DISCRETE>** or **<LINEAR>**. Both tags require an arbitrary number of pairs of values in the form (position, temperature), e. g.

```
<TPROFILE>
  <DISCRETE>
    0.000    300    # z-position [m], temperature [K]
    0.001    400
    0.002    450
    0.003    500
  </DISCRETE>
</TPROFILE>
```

Using **<DISCRETE>** a piecewise constant temperature profile is defined. The position indicates the begin of the interval with the specified temperature. Define only one pair of values for a constant wall temperature.

<LINEAR> is used to define a piecewise linear temperature profile. Specify at least two pairs of values in that case.

The tag **<ISOTHERM>** is used for multiple runs of DETCHEM^{CHANNEL} under isothermal conditions. In that case the wall temperature and the inlet temperature are set to a constant value. Also the inlet velocity is being adjusted to the new inlet temperature automatically. Specify a temperature for each run, e. g.

```
<TPROFILE>
  <ISOTHERM>
    300    # temperature [K]
    400
    500
    600
  </ISOTHERM>
</TPROFILE>
```

In this example DETCHEM^{CHANNEL} will perform four simulations at temperatures 300, 400, 500, and 600 K, respectively. The number of the output file is incremented automatically.

No temperature profile needs to be defined in case of adiabatic conditions. Activate the **adiabatic** member instead. Note: **adiabatic** suppresses heat exchange through the channel wall and therefore should not be used in conjunction with a superordinate simulation like DETCHEM^{MONOLITH}.

Especially in the case of adiabatic calculations, the reaction may not ignite under the given conditions. The option **ignite_at** can be used to simulate a spark to initiate the chemical reaction. The temperature specified in this option is used as a minimum temperature for the calculation of the chemical source terms. All properties of the fluid are calculated at the actual temperature. The **ignite_at** temperature shall be chosen high enough to initiate the chemical reaction, but it should be lower than the expected reaction temperature.

In case a reaction mechanism is negligible for low temperatures, the option **ignore_below** can be used to speed up calculations. For temperatures lower than the specified value, the chemical source terms will not be calculated and are set zero.

13.3 Output

Depending on the settings in the **<OUTPUT>** section of the input file, DETCHEM$^{\text{CHANNEL}}$ produces screen output and file output.

13.3.1 Screen output

If monitoring is activated, the screen output will look like below

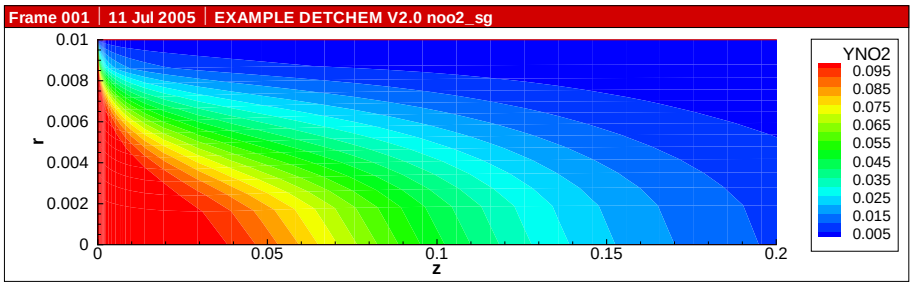
```
************************************************************************
****                       DETCHEM CHANNEL                          ****
****                       STEFFEN TISCHER                          ****
****                  VERSION 2.0.001 ** 2003/07/24                 ****
************************************************************************
     0     0.00000000     0.01000000   1123.00   1123.00
     1     0.00000000     0.01000000    300.00   1123.00
     2     0.00000000     0.01000002    300.00   1123.00
     3     0.00000000     0.01000014    300.00   1123.00
...
   161     1.00000000     0.01000000   1123.03   1123.00
 Iterations: 161    Errors: 0
```

The five colums show the number of integration step, the z-position along the channel, covergence monitoring for the radial coordinate (should always be close to **rmax**), temperature at the channel center, and temperature of the wall.

13.3.2 *outg***.plt*

The file ***outg***.plt*** contains two-dimensional data of the gas-phase of the channel. The file is in TECPLOT format and can be used to draw two-dimensional contour plots. Since the data is stored in ASCII format, it can easily be imported in to other programs, e. g. Microsoft Excel (but Excel cannot be used to draw 2d-contour plots).

The variables in ***outg***.plt*** are: axial coordinate (z), radial coordinate (r), temperature (T), axial velocity (u), radial velocity (v), density (rho), dynamic pressure (pdyn), either mole or mass fractions (prefix X or Y respectively – depends on settings in **<OUTLET>** section).



13.3.3 *outs***.plt*

In case of surface chemistry, the file ***outs***.plt*** is written. The file is in TECPLOT format and can be used to draw one-dimensional line plots of the variables at the wall. The data can easily be imported in to and displayed with other programs, e. g. Microsoft Excel.

The variables in ***outs***.plt*** are: axial coordiante (z), radial coordinate (r – shows convergence of the solver), wall temperature (T), surface coverages (species name without prefix), gas-phase mole or mass fractions (prefix X or Y respectively – depends on settings in **<OUTLET>** section).

13.3.4 *outflux***.plt*

If the user wants to get information about the averaged conversion and selectivity at each position along the channel axis, the file ***outflux***.plt*** can be used. The file contains the radially avaraged mole or mass fractions vs. the axial coordinate. The data is stored in TECPLOT format, but it can easily be imported in to and be displayed with other programs, e. g. Microsoft Excel.

13.3.5 *summary_out.dat*

Each run of DETCHEM^{CHANNEL} appends one line in the file ***summary_out.dat***. If the file does not exist, it will be created and a header line is written first. In ***summary_out.dat*** the inlet and the outlet compostions of each channel simulation are summarized. Hence it can be used to compare conversion and selectivity of multiple runs of DETCHEM^{CHANNEL}.

The file ***summary_out.dat*** is a simple text file that has been designed to be opened with any spreadsheet program, e. g. Microsoft Excel. It contains the file numbers, velocities, temperatures and eiter mole or mass fractions of all gas-phase species at the inlet (suffix i) and outlet (suffix o). The header is only valid as long as the species list does not change. So remember to delete or move ***summary_out.dat*** manually when the species list has changed.

13.4 Examples

There are a few examples supplied along with the installation. Each example directory contains the following files: ***channel.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism file, a surface mechanism file, ***thermdata***, and ***moldata***. The optional washcoat input has been saved in a separate file ***washcoat.inp***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

13.5 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.

When setting up a new case, we also suggest to start as simple as possible and to become more complex gradually. Check if the geometry settings and wall temperature profile does not contain errors by running simulations without chemistry first. Then add surface and gas-phase mechanisms one after another. Always check for error messages and watch the solution process to detect convergence problems. In case of problems, adjusting the solver and tolerance parameters may improve the performance.

13.6 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***channel***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/channel*** and call ***./channel***.

The latter option does not require setting of system variables and is therefore less system specific.

13.7 BLAYER version

DETCHEM$^{\text{BLAYER}}$ is a software package for simulation of 2-D chemically reacting flows in cylindrical channel, which is similar to DETCHEM$^{\text{CHANNEL}}$ but using different numerical methods. The PDE (governing equations) are semi-discretized using a method of lines leading to a special structure DAE system, which is solved using a special tailored BDF code developed based on the DAESOL code (for more details see H. D. Minh, PhD thesis, Department of Mathematics and Computer Science, University of Heidelberg, 2005).

The input and output files have similar formats as of DETCHEM$^{\text{CHANNEL}}$. In the following we only discuss the differences.

13.7.1 User Input

In the **<SOLVER>** tag, the user can specify the following name-value pairs, only require names have to be specified, optional names can be specified, if they are not specified the default value will be taken.

```
<SOLVER>
   hini=1.d-12           !optional, default 1.d-12
   hmax=0.025            !optional, default 1.d-1
   relax-jac = yes       !optional, default yes
   onTLFD = yes          !optional, default yes
   eta1  = 1.d-3         !optional, default 1.d-3
   eta2  = 1.d-8         !optional, default 1.d-8
   max-bdf-order = 5     !optional, default 5
   fd-order = 1.d0       !optional, default 1
</SOLVER>
```

Usually, the default values work fine. However, in the case of failure of the solver one can try to change the **eta1** and **eta2** parameters (type: double). The recommended range of **eta1** would be $1.d-1 \rightarrow 1.d-8$, the recommended range of **eta2** would be $1.d-4 \rightarrow 1.d-13$.

The option **relax-jac** (yes/no) is used for specifing whether the computation of Jacobian by finite difference methods in the BDF code should be relaxed or not. By default, this option is turned on (**relax-jac = yes**), this would improve performance of the code (many times faster!).

The option **max-bdf-order** (type: integer) is used for specifing the maximum order of the BDF methods used in the BDF code. By default, it is set to 5. The valid range of **max-bdf-order** would be $1 \rightarrow 5$.

The option **fd-order** (type: double) is used for specifing how the radial grid is generated. It specifies how the radial grid points are distributed. The higher it is, the more the radial grid is fine near the surface. The recommended value would be between 1.0 and 3.0.

In the **<TOLERANCE>** tag, the user can specify the following name-value pairs

```
<TOLERANCE>
   atol=1.d-10      ! optional, absolute tolerance, default 1.d-9
   rtol=1.d-3       ! optional, relative tolerance, default 1.d-3
</TOLERANCE>
```

The user can specified multiple $F_{cat/geo}$ profiles for different surface sites.

```
<FCATGEOPROFILES>
  Nsites = 1     ! number of surface sites
<list> Pt  </list>
#  z      fcat/geo
  0.0     15
  0.0254  25
</FCATGEOPROFILES>
```

13.7.2 Output

The screen output looks like:

```
*************************************************************************
****                        DETCHEM BLAYER                           ****
****                        HOANG DUC MINH                           ****
****                  VERSION 2.0.1 ** 2007/07/16                    ****
*************************************************************************
    z:   1.0000E-11  step_size:  1.0000E-11     order:  1 next_order: 1
    z:   2.0000E-11  step_size:  2.0000E-11     order:  1 next_order: 1
    z:   4.0000E-11  step_size:  4.0000E-11     order:  1 next_order: 1
    z:   8.0000E-11  step_size:  8.0000E-11     order:  1 next_order: 1
    z:   1.6000E-10  step_size:  1.6000E-10     order:  1 next_order: 1
    z:   3.2000E-10  step_size:  3.2000E-10     order:  1 next_order: 1
    z:   6.4000E-10  step_size:  6.4000E-10     order:  1 next_order: 1
    .         .          .          .           .     .     .       .
    z:   1.4971E-02  step_size:  2.6213E-03     order:  3 next_order: 3
    z:   1.7592E-02  step_size:  3.0224E-03     order:  3 next_order: 3
    z:   2.0615E-02  step_size:  3.4702E-03     order:  3 next_order: 3
    z:   2.5400E-02  step_size:  2.2096E-03     order:  3 next_order: 3
```

```
=====================================================================
                          SIMULATION IS SUCCESSFULLY FINISHED
                                          NUMBER OF STEP:  78
       THE OUTPUT IS THE bl.plt FILE FOR GAS PHASE SPECIES AND
                        THE surf.plt FILE FOR SURFACE SPECIES
=====================================================================
```

The gas phase species mass fractions/mole fractions and the temperature, the pressure, the axial velocity profiles are written in the file ***bl.plt***, in 2-D TECPLOT format.

The average of gas phase species mass fractions/mole fractions in radial direction are written in in the file ***avg.plt***, in 1-D TECPLOT format.

The the surface species coverages are written in the file ***surf.plt***, in 1-D TECPLOT format.

Chapter 14

DETCHEM^{GRIDGEN3D}

14.1 Introduction

DETCHEM^{GRIDGEN3D} is a preprocessor for generating grid information used by DETCHEM^{MONOLITH} in case of 3-dimensional simulations. Run DETCHEM^{GRIDGEN3D} prior to setting up your 3d-problem with DETCHEM^{MONOLITH}. The preprocessor can be used to create simple grids for an arbitrary cross-section or to import grids in TECPLOT format that have been created with other programs. If the user wishes to run a 3d-simulation with spatially varying inlet conditions - the grid of the inlet data must be imported using DETCHEM^{GRIDGEN3D}.

In any DETCHEM^{MONOLITH} simulation the cross-section of the domain must not change along the channel axis. Therefore it is sufficient to specify the geometry of the cross-section. This arbitrary two-dimensional shape will be discretized by a triangular grid. DETCHEM^{GRIDGEN3D} tries to make the grid as regular as possible, so that neighboring cells have similair size.

On execution, DETCHEM^{GRIDGEN3D} generates an output file ***r_grid.inp*** that contains the radial grid specification to be included in DETCHEM^{MONOLITH}'s input file ***monolith.inp***.

14.2 User Input

DETCHEM^{GRIDGEN3D} requires only one input file named ***gridgen.inp***. An example is shown below.

```
<GRIDGEN>

  <POLYGON>
    name="poly"
    n=20          # number of points
    r=0.10        # radius
    dmax=0.03     # maximum distance between 2 grid points
  </POLYGON>

  <LAYER>
    name="insu"
    cells=2
    thickness=0.005
  </LAYER>

  <BORDER>
    name="bound"
  </BORDER>

</GRIDGEN>
```

Since DETCHEM^{GRIDGEN3D} is a preprocessor and does not do any simulation, there is no species or mechanism information in the input file. ***gridgen.inp*** immediately starts with the **<GRIDGEN>** tag that may contain the following subsections:

<**POLYGON**>	define a regular polygon
<**CIRCUMPOINTS**>	define an arbitrary shape
<**TECPLOT**>	use a grid in TECPLOT format
<**LAYER**>	define additional layers
<**BORDER**>	define border

Depending on the type of grid definition, one of the tags **<POLYGON>**, **<CIRCUMPOINTS>**, or **<TECPLOT>** is required. For all types, additional layers may be declared in the **<LAYER>** subsection. Finally, a border name declaration is required.

14.2.1 The <POLYGON> tag

Use this tag, if the cross-section of the monolith has a shape that can be fitted by a regular polygon with an ellipsoidal circumference. Hence, regular triangles, squares, hexagons and other polygons can be modeled. The **<POLYGON>** sections expects the following parameters.

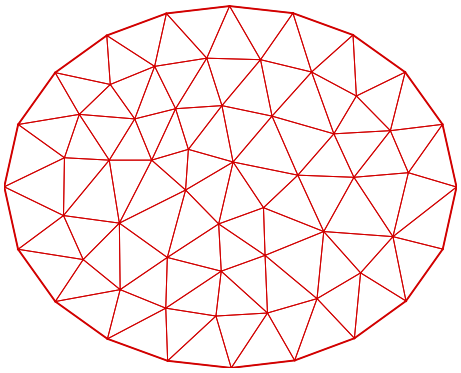
n=*i*	number of edges
r=*d*	radius of polygon [m]
r2=*d*	second radius of an ellipsoidal polygon [m] (default: r)
dmax=*d*	maximum distance of grid points [m] (default: circumference / n)
name=*s*	layer name (used as ID)

Define the shape, maximum grid size and a name of the polygomial domain. The **name** member (use up to 8 characters) will be used in the grid definition of the DETCHEM^MONOLITH input file to refer to this domain.

The definition

```
<POLYGON>
  name="poly"
  n=20          # number of points
  r=0.10        # radius
  r2=0.08       # 2nd radius (if elliptical) - can be omitted
  dmax=0.03     # maximum distance between 2 grid points
</POLYGON>
```

results in a grid like shown below.



14.2.2 The <CIRCUMPOINTS> tag

In this section, an arbitrary shape of the monolithic domain can be declared by giving the coordinates of its vertices. Besides the coordinates two more parameters are required:

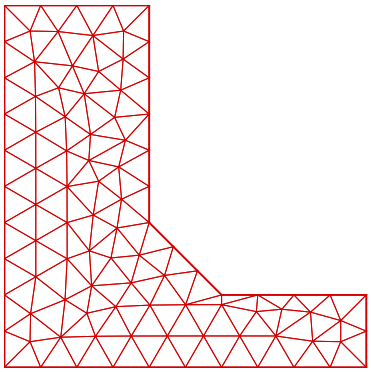
dmax=*d*	maximum distance of grid points [m]
name=*s*	layer name (used as ID)
x-coordinate y-coordinate	coordinates (x,y) of circumferential points [m]

The **name** member (use up to 8 characters) will be used in the grid definition of the DETCHEM^MONOLITH input file to refer to this domain.

The definition

```
<CIRCUMPOINTS>
   name="poly"
   dmax=0.01       #maximum distance between two grid points
    0       0      #y- and z-coordinates of circumference
    .10     0
    .10     .02
    .06     .02
    .04     .04
    .04     .10
    0       .10
</CIRCUMPOINTS>
```

results in a grid like shown below.



14.2.3 The <TECPLOT> tag

If the user wants to use a grid created by other applications, then the **<TECPLOT>** tag shall be used. Especially in case of spatially varying inlet conditions the grid of the inlet data must be the same as the grid of the monolith's cross-section. Thus derive the monolith grid from the inlet data grid using the **<TECPLOT>** tag.

An example is shown below

```
<TECPLOT>
  name="poly"
  file="velin001.dat"   # name of tecplot file
  3d-slice=y            # transformation into y-z-plane necessary
</TECPLOT>
```

where the possible options are

file=*s*	TECPLOT file name
3d-slice=*b*	co-ordinates are given in 3d (yes) or 2d (no) – if yes transformation to 2d is done (default: no)
name=*s*	layer name (used as ID)

The **name** member (use up to 8 characters) will be used in the grid definition of the DETCHEM$^{\text{MONOLITH}}$ input file to refer to this domain.

If the **3d-slice** option is not set, the first two variables of the TECPLOT file will be interpreted as the coordinates of the grid points. Otherwise, The first three variables are considered and the coordinates are transformed into the y-z-plane automatically.

14.2.4 The <LAYER> tag

This tag may be used to define additional layers around the monolithic domain, e.ġ. to represent insulation layers. These layers must have equidistant thickness around the whole domain.

Use a new **<LAYER>** tag for each layer with different material properties. The options within this section are

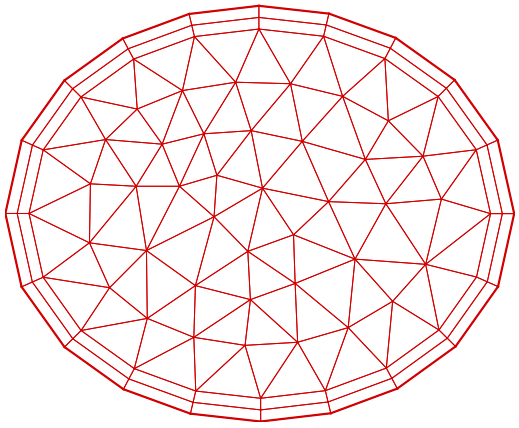
thickness=*d*	layer thickness [m]
cells=*i*	number of cells (default: 1)
name=*s*	layer name (used as ID)

The domain and the layers must have different names (8 significant characters), because these names are used in the grid declaration of DETCHEM$^{\text{MONOLITH}}$.

If the following definition is added to the example shown in section 14.2.1

```
<LAYER>
  name="insu"
  cells=2
  thickness=0.01
</LAYER>
```

the result is



14.2.5 The <BORDER> tag

In order to assign a boundary condition to the whole domain (including additional layers), its border needs an ID that is used in the input file of DETCHEM$^{\text{MONOLITH}}$. This ID is declared as shown in the example below.

```
<BORDER>
  name="bound"
</BORDER>
```

Just assign a string of up to 8 characters to the **name** member.

14.3 Output

DETCHEM$^{\text{GRIDGEN3D}}$ produces two output files ***r_grid.inp*** and ***r_grid.plt***. The former contains the grid information to be read by DETCHEM$^{\text{MONOLITH}}$, and the latter is a TECPLOT file for visualization of the generated grid.

14.3.1 Screen output

There is only few screen output when running DETCHEM$^{\text{GRIDGEN3D}}$. The program informs the user about the current step and eventual errors. After successful execution of the program the screen looks like follows

```
Initialize Polygon
Triangulate Polygon
Initialize Cells
Add layer
Calculate nodes weight
Calculate flux parameters
Calculate cell areas
Write grid
```

14.3.2 *r_grid.inp*

The file ***r_grid.inp*** will be necessary to set up a 3d simultion with DETCHEM$^{\text{MONOLITH}}$. This file should not be edited or changed by the user. Just copy it to your DETCHEM$^{\text{MONOLITH}}$ case directory and include the information in the input file ***monolith.inp***. Details will be explained in chapter **??**.

14.3.3 *r_grid.plt*

Before using the grid with DETCHEM$^{\text{MONOLITH}}$ it can be visualized using the file ***r_grid.plt***. This file contains the grid information in TECPLOT format. Upon loading the file as 2d plot with TECPLOT the grid is displayed. The uniformity of the grid can be visulized by activating a contour plot of the cell volumes.

Since the file is saved in text format, it can also be imported into other viewing programs. There are three variables: the two coordintes and the cell volume. They are stored in block format, i.e. all values of one variable sequentially. Below, the cell definitions follow. Four integers make one cell. If the first and the last point are identical, the cell is triangular, otherwise it is quadrilateral.

14.4 Examples

There is an example for each of the three types of grid definition in the distribution. In each subdirectory you can find a file ***gridgen.inp***. In the case of the TECPLOT import example there is also another file ***velin001.dat***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

14.5 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***gridgen3d***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/gridgen3d*** and call ***./gridgen3d***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 15

DETCHEM$^{\text{RESERVOIR}}$

15.1 Introduction

DETCHEM$^{\text{RESERVOIR}}$ serves as a transient wrapper for single channel simulations with storage effects. It can be used in conjunction with DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$. Unlike in DETCHEM$^{\text{MONOLITH}}$, the wall temperature profile of the channel does not change in time. Only the concentrations of selected surface species (those of the storage medium) are transient. For the storage medium we must assume that the rate of reaction is slow enough to be decoupled from the flow field. That is, the time scale for storage reactions must be larger than the residence time of the gases inside the channel. For surface species involved in these reactions, DETCHEM$^{\text{RESERVOIR}}$ solves the following transient species equation

$$\frac{\mathrm{d}c_i^{\text{storage}}}{\mathrm{d}t} = \dot{s}_i(c^{\text{gas}}, c^{\text{surf}}, c^{\text{storage}}) \tag{15.1}$$

The concentrations of the gas-phase species c^{gas} are calculated by a quasi steady-state channel simulation using DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$, whereas for the non-storage surface species c^{surf} steady-state is also assumed. Thus, for each time step DETCHEM$^{\text{RESERVOIR}}$ calls DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$ with fixed storage concentrations c^{storage}. After the single channel simulation, the locally resolved gas-phase concentrations c^{gas} are passed back to the reservoir model. Then equation 15.1 is integrated for a given time step and the cycle starts from the beginning again. The principle of the decoupling of the two codes is illustrated in figure 15.1.

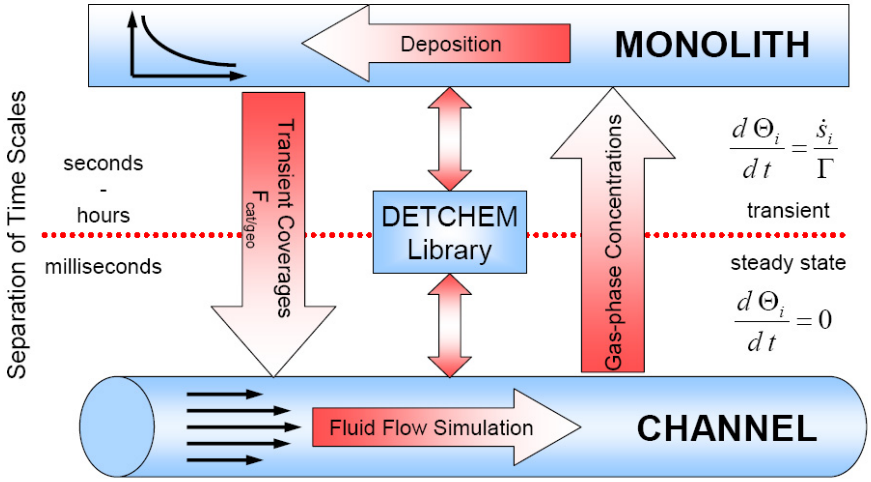


Figure 15.1: Schematic communication between DETCHEM$^{\text{RESERVOIR}}$ and single channel simulation

15.2 User input

The input required for DETCHEM$^{\text{RESERVOIR}}$ is similar to DETCHEM$^{\text{MONOLITH}}$, however a few sections can be dropped due to the simplifications of the model. DETCHEM$^{\text{RESERVOIR}}$ requires one input file named ***reservoir.inp***. An example is shown below.

```
{include channel.inp}

 <RESERVOIR>

 <GRID>
   <XDOMAIN>
     from=  0.
     to=  200.d-3
     cells= 40
     aspratio=  1.
   </XDOMAIN>
 </GRID>

 <SOLUTION>
   time =  130
   h = 2.6
   T/K = 623
 </SOLUTION>

 <OUTPUT>
  <do> from 5 to 55 step 5 </do>
  59.9
  60.1
  64.9
  <do> from 65 to 125 step 5 </do>
  write_restart=yes
 </OUTPUT>

 <STORAGE>
   surface="BaCO3"
   mechanism="Speicher"
 </STORAGE>

 <INLET type=variable>
   file=inlet.inp
 </INLET>

 </RESERVOIR>
```

The file ***reservoirinp*** must start with the definitions necessary for the channel calculation. This implies that first, the species definiton, the mechanism definition, the optional washcoat deinition, and the channel definition appear first in the input file. It is probably most convenient to use the same input file as for the channel simulations and include the channel input file by an **include** statement. The DETCHEM$^{\text{RESERVOIR}}$ specific input data follows enclosed in **<RESERVOIR>** tags. The options inside this section are listed in the following table.

Version=*s*	input file version (not evaluated)
<**GRID**>	define grid (must be the first tag)
<**SOLUTION**>	define solver options
<**OUTPUT**>	define output options
<**POSTPROCESS**>	postprocessing
<**STORAGE**>	storage model
<**INLET**>	define inlet conditions

<GRID> – grid definition

For the calculation of the transient concentraition profiles, the channel needs to be discretized in axial direction. This is done by the (one-dimensional) grid definition. The **<GRID>** tag must contain one or more **<XDOMAIN>** sections. In most cases, one such section should be enough. However, if the grid should be non-uniform, the use of more than one **<XDOMAIN>** definitions may be useful. The **<XDOMAIN>** section can also be used to override the default temperature and catalyst distribution ($F_{\text{cat/geo}}$) in this section. e. g.

```
<GRID>
  <XDOMAIN>
    from    =  0
    to      =  0.02
    cells   = 10
    aspratio=  1.05    # non-uniform grid at entrance
  </XDOMAIN>
  <XDOMAIN>
```

```
    to      =  0.1
    cells   = 20
    T/K     = 500      # override temperature from <SOLUTION>
    <Fcatgeo>          # override Fcat/geo from <SURFACE-MODEL>
       BaCO3   20
    </Fcatgeo>
  </XDOMAIN>
</GRID>
```

Each domain can be divided into several cells. In each cell represents one set of transient concentrations. Therefore, the number of cells should be chosen accordingly to the expected concentration gradients. The user must further make sure that the total length of the discretized grid (the **to** parameter in the last domain) equals the length of the channel. The members in each **<XDOMAIN>** tag are:

from=*d*	axial coordinate of start of the domain (default: end of the previous domain)
to=*d*	axial coordinate of end of the domain
cells=*i*	number of grid cells
aspratio=*d*	aspect ratio, i.e. ratio of the lengths of two adjacent cells (default: 1)
T/K=*d*	wall temperature (overriding definition from **<SOLUTION>**)
<Fcatgeo>	catalyst composition (overriding definition from **<SURFACE-MODEL>**)

<SOLUTION> – operation parameters

The operation conditions of the DETCHEM$^{\text{RESERVOIR}}$ simulation are defined within the **<SOLUTION>** tag. These parameters are the total time of simulation, the maximum step size and the (isothermal) wall temperature. Alternatively to a constant wall temperature, a linear temperature profile can be defined. Moreover, if a previous simulation is to be continued, you can use an exisiting output file for initialization.

All available options are summarized below:

time=*d*	end time [s]
h=*d*	integration time step [s]
T/K=*d*	temperature [K]
T_in=*d*	temperature at inlet [K]
T_out=*d*	temperature at outlet [K]
dT/s=*d*	temperature ramp for TPD simulations [K/s]
dT/min=*d*	temperature ramp for TPD simulations [K/min]
restart=*s*	restart file name

<OUTPUT> – output options

Several output files are written during a DETCHEM$^{\text{RESERVOIR}}$ simulation. Use the **<OUTPUT>** tag to define the time steps for wich individual output files are desired. The time steps can be listed individually (just give the numeric value of the time in seconds) or generated with regular frequency (use a **<DO>** tag). The times listed must be in ascending order.

Furthermore, there is an option to generate an output file ***restart.plt*** that contains the latest status of the simulation. This file might be useful in case a simulation needs to be restarted after an interruption.

The option within the **<OUTPUT>** tag can be summarized as

<**DO**>	generate equistant output times
output time	time for that output shall be written [s]
WRITE_RESTART=*b*	write a file restart.plt after each iteration (default: no)

and the **<DO>** parameters are

from=*d*	start time [s]
to=*d*	end time [s]
step=*d*	time between two outputs [s]

<STORAGE> – transient species and mechanism

The main purpose of simulations with DETCHEM$^{\text{RESERVOIR}}$ is the storage process. The user must choose one surface type for which the transient concentrations are calculated. For instance, when a catalytic converter contains platinum and barium as catalytic active components, the platinum surface may be considered in quasi steady state, whereas the slow NOx storage processes occur on the barium sites. In this case, choose the barium surface as storage surface (use the name of the surface as it is defined in the **<SPECIES>** section at the beginning of the input file). The reaction mechanism containing the relavant reactions for the storage must also

be referenced. In most cases this mechanism can be identical with the complete surface reaction mechanism. Refer to it by the name as it is defined in the **<MECHANISM>** section at the beginning of the input file.

surface=*s*	name of the storing surface
mechanism=*s*	name of the storage mechanism

<INLET> – inlet information

The transient nature of DETCHEM^{RESERVOIR} will usually require transient input data. However, it is also possible to define constant input data directly inside this input file.

In the latter case, just specify the inlet velocity and the species composition, e. g.:

```
<INLET type=constant>
  u0=  2.            # velocity in m/s
  <MOLEFRAC>
    NO2       0.1    # mole fraction
    N2        *      # balance
  </MOLEFRAC>
</INLET>
```

As an alternative to mole fractions, mass fractions can be defined using the tag **<MASSFRAC>**.

In most cases, however, the inlet conditions will vary in time. This is declared like in the following example:

```
<INLET type=variable>
  file     = inlet.inp   # name of the external file
  velfactor= 2.0         # velocity conversion factor
</INLET>
```

Instead of declaring all information inside this input file, the transient inlet information is stored in an external file, e. g. ***inlet.inp***. The format of this external file will be described later in this chapter. Sometimes, the inlet velocity needs to be converted due to changes in the diameter of the channel or in the temperature. Therefore the member **velfactor** can be assigned a numerical value. The velocities in the external file are multiplied with this factor. Note: Use for convenience this conversion factor in case your external file contains mass or volume fluxes instead of velocities.

All inlet data can also be altered by user defined functions. If the user wants to provide own code (see chapter **??** for DETCHEM^{MONOLITH}), activate the **user** option.

Thus, the inlet options can be summarize as follows:

type=*s*	type of inlet condition: constant or variable
FILE=*s*	name of inlet input file
velfactor=*d*	conversion factor for inlet velocities (default: 1)
user=*b*	use user defined inlet conditions (default: no)

External inlet file

For variable inlet conditions it is necessary to prepare another input file. It can have any name (e. g. ***inlet.inp***). Use this name as reference for the **file** member in the **<INLET>** section of the input file.

The inlet file consists of two sections: **<INLETINFO>** and **<INLETDATA>**. An example is shown below

```
<INLETINFO>
  u = 0.8       # constant velocity
  <MOLEFRAC list>
     CH4
     O2
     N2
  </MOLEFRAC>
</INLETINFO>

<INLETDATA>
#time   CH4    O2    N2
    0   0.0   0.2   0.8
    5   0.1   0.2   0.7
   60   0.1   0.1   0.8
  120   0.1   0.1   0.8
</INLETDATA>
```

In **<INLETINFO>** the user needs to specify which data is read from a list or file and which is constant. Assign a constant value or the keyword **list** to the parameter **u** for velocity and to **<MASSFRAC>** or **<MOLEFRAC>** for the species composition.

The section **<INLETDATA>** contains the values of all data that has been assigned the **list** option. Although it is not required, it is strongly recommended to use a column oriented format for better readability. The first

column contains the start time of the following conditions. These conditions are valid until the next time in the list or until the simulation ends. The following columns contain the velocity and species composition in that order. If they are constant, the column is skipped.

In **<INLETDATA>** the user also may specify further options between two data lines (not before the first data line)

velocity	velocity [m/s]
mass or mole fraction	mass or mole fractions
dtmax=d	maximum step size (use in case step size shall vary with inlet data) (default: difference to next time in list)
skip_failed=b	skip inlet data in case it produces erroneous results (default: no)
<**SURFACE-MODEL**>	change <SURFACE-MODEL> options

The parameter **dtmax** may be useful in case of time dependent step size, for instance if there is a cycle with a long and slow storage phase and a short and fast regeneration phase. The binary option **skip_failed** may be activated in case the simulation is not successful for a few sets of inlet data. If activated, the inlet data of unsuccessful simulation time steps is skipped. The simulation continues with the next time step. Although the result is erroneous at this particular time step, the simulation may continue instead of abortion due to an error.

In case of the use of the washcoat effectiveness factor model, it may be required to change the effectiveness factor species according to the current inlet data. For this purpose the settings of the **<SURFACE-MODEL>** section (see chapter 3.8) may be altered by the transient inlet file. This can be done by including a **<SURFACE-MODEL>** definition after the data of the according time step. Most conveniently this can be done by saving the different **<SURFACE-MODEL>** sections in a separate files and including them in the input file, e.ġ.

```
<INLETINFO>
  u = 0.8       # constant velocity
  <MOLEFRAC list>
     CO  NO  O2  N2
  </MOLEFRAC>
</INLETINFO>

<INLETDATA>
#time   CO    NO    O2     N2
    0   0.01  0     0.09   0.9
 {include surface-model1.inp}  # CO is limiting species
   60   0.09  0.01  0      0.9
 {include surface-model2.inp}  # NO is limiting species
</INLETDATA>
```

15.3 Output

*reservoir***.plt*

At each time defined in the **<OUTPUT>** section a file ***reservoir***.plt*** is written. The file is in TECPLOT format and can be used to draw one-dimesional contour plots of the storage concentrations. It can also be imported into other spreadsheet programs like Microsoft Excel.

The file header contains the time when the file was written. The user may change this time when this file shall be used as a restart file of a new simulation.

In contrast to ***outs***.plt*** files, the storage concetrations are now written with respect to $F_{\mathrm{cat/geo}}$. Since the capacity may change along the channel, the written coverages are already multiplied with $F_{\mathrm{cat/geo}}$ of the storage component.

reservoir.plt

This file contains the data of all ***reservoir***.plt*** files. The file is in TECPLOT format and can be used to draw two-dimesional contour plots of the storage concentrations depending on time and position.

Note: In case of a restart, the previosly written data may be lost. Therefore it is advisible to copy ***reservoir.plt*** to a different file and concatenate the data manually after successful simulation.

restart.plt

The file ***restart.plt*** contains the same data as a ***reservoir***.plt*** file. If the **write_restart** option is activated, the file will be written in each time step. Therefore it always contains the state of the simulation at the latest time step.

global.plt

The file ***global.plt*** summarizes the global inlet and global outlet composition at each time step (not only the times listed). The file is in TECPLOT format and can be used to draw line graphs of gas-phase temperature and mass fractions. However, this format also allows for simple import into other spreadsheet programs (e. g. Microsoft Excel).

The variables in this file are: time, inlet temperature, inlet mass fractions, outlet temperature, outlet mass fractions, and an error flag. You can use this file for simple calculation of conversions and selectivities.

It is recommended to pay attention the the error flag. If the flag is not zero, an error in one or more channel simulations at this time step has occurred. The outlet composition at this time step is most likely wrong. If this only happens sparsely, you may delete the wrong line. However, if the error flag keeps to be set, the simulation was not successful.

A new file ***global.plt*** is written on each execution of DETCHEM$^{\text{RESERVOIR}}$. If you want to restart an earlier calculation, save ***global.plt*** with a different name before calling the executable again.

Channel output

For DETCHEM$^{\text{RESERVOIR}}$ it is not required to do post processing in order to generate channel output. All channel output files are written automatically for each time step listed in the **<OUTPUT>** section of the input file.

15.4 Examples

There are several examples in the distribution. Each directory contains several input files (****.inp***), species and mechanism information as well as the databases ***thermdata*** and ***moldata***. The ***.plt*** files are generated on execution.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

15.5 Running the tool

There are two executables for DETCHEM$^{\text{RESERVOIR}}$ in the ***($DETCHEM_DIR)/bin*** directory. They are named ***reservoir_channel*** and ***reservoir_plug*** indicating the channel model used for the single channel simulation, i. e. DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$, respectively.

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***reservoir_channel*** or***reservoir_plug*** .
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/reservoir_channel*** and call ***./reservoir_channel***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 16

DETCHEM^{SOFC}

16.1 Introduction

DETCHEM^{SOFC} is a quasi two dimensional numerical model for the simulation of solid-oxide fuel cells (SOFC). The model can be applied to a) button cell, b) planar cell co-flow configurations, and c) tubular cell co-flow configuration. The model takes into account of the details of transport as well as chemical/electrochemical processes. The thermo-catalytic processes occurring in the cell can be represented by a global reaction mechanism or by an elementary reaction mechanism. The examples following this distribution utilizes an elementary like reaction mechanism for the steam reforming of methane on Ni catalysts. The electrochemical processes are implemented by a modified Butler-Volmer equation. The transport processes within the porous electrodes are modeled by the Dusty-Gas-Model. (DGM).

16.2 Physical and chemical fundamentals

16.2.1 Heat and mass transport

Channel Flow

Flow through fuel and air channels is assumed to be one dimensional and laminar in nature. The plug flow equation for species continuity in the channel are given by

$$\frac{\partial(\rho_f Y_k)}{\partial t} = -\frac{\partial(\rho_f u Y_k)}{\partial z} + \frac{P_\mathrm{e}}{A_\mathrm{c}} J_k W_k, \quad k = 1, \ldots, K_g. \tag{16.1}$$

The velocity is calculated from the momentum balance equation

$$\frac{\partial \rho_f u}{\partial t} = -\frac{\partial(\rho_f u u)}{\partial z} + u \sum_{k=1}^{K_g} \frac{P_\mathrm{e}}{A_c} J_k W_k. \tag{16.2}$$

Here P_e is the perimeter associated with the membrane electrode assembly, and A_c is the cross sectional area of the channel. Assuming constant pressure in the channels density is calculated from the ideal gas equation

$$\rho_f = \frac{p\bar{M}}{RT} \tag{16.3}$$

In Eq. 16.1 and 16.2 J_k is the flux at the electrode channel interface and is calculated using dusty gas model (Eq. 16.11). Temperature in the flow channels is determined from the energy equation

$$\frac{\partial(\rho_f C_{pf} T_f)}{\partial t} = -\frac{\partial(u \rho_f C_{pf} T_f)}{\partial z} - \frac{h}{D_\mathrm{h}}(T_f - T_s). \tag{16.4}$$

The heat transfer coefficient h is evaluated from Nusselt number

$$Nu = \frac{h D_h}{k}, \tag{16.5}$$

which is expressed empirically as

$$Nu = 3.095 + 8.933\left(\frac{1000}{Gz}\right)^{-0.5386} \exp\left(-\frac{6.7275}{Gz}\right) \tag{16.6}$$

where Graetz number Gz is given by

$$Gz = \frac{D_h}{z} RePr. \tag{16.7}$$

Porous media transport

Species transport through the porous media is assumed to be one dimensional along the thickness of the porous structure and is given by

$$\frac{\partial(\epsilon\rho_f Y_k)}{\partial t} = -\frac{\partial(J_k W_k)}{\partial y} + \dot{s}_k W_k A_s. \tag{16.8}$$

Where $\dot{s}_k$ is the heterogeneous production rate of the chemical species k and A_s is the specific catalyst area available for surface reactions. The total density of the gas-phase within the porous structure can be calculated from

$$\frac{\partial(\epsilon\rho_f)}{\partial t} = -\sum_{k=1}^{K_g} \frac{\partial(J_k W_k)}{\partial y} + \sum_{k=1}^{K_g} \dot{s}_k W_k A_s. \tag{16.9}$$

In the above equations the fluxes J_k are evaluated using DGM.

Dusty-Gas-Model

The model accounts for three different transport mechanisms, molecular diffusion, Knudsen diffusion, and viscous transport. The total diffusive flux in DGM results from molecular diffusion acting in series with Knudsen diffusion. The viscous porous media flow (Darcy flow) acts in parallel with diffusive flux. The DGM can be written as an implicit relationship among molar concentrations, fluxes, concentration gradients and pressure gradient as

$$\sum_{l\neq k} \frac{[X_l]J_k - [X_k]J_l}{[X_T]\mathcal{D}^e_{kl}} + \frac{J_k}{\mathcal{D}^e_{k,kn}} = -\nabla[X_k] - \frac{B_g[X_k]}{\mu\mathcal{D}^e_{k,kn}}\nabla p. \tag{16.10}$$

Eq. 16.10 can be inverted to develop a direct formulation of the diffusive molar flux vector J_k as

$$J_k = -\left[\sum_{l=1}^{K_g} \mathcal{D}^{\mathrm{DGM}}_{kl}\nabla[X_l] + \left(\sum_{l=1}^{K_g} \frac{\mathcal{D}^{\mathrm{DGM}}_{kl}[X_l]}{\mathcal{D}^e_{l,kn}}\right)\frac{B_g}{\mu}\nabla p\right]. \tag{16.11}$$

16.3 User Input

Before running DETCHEM$^{\mathrm{SOFC}}$the user must prepare an input file ***sofc.inp***. An example is shown below.

```
{verbose y}
{include species.inp}
{include mech.inp}
<SURFACE-MODEL>                      # surface chemistry model
 <CHEMSURF>
   hini=1.0d-10                      # initial time for chemsurf integration
time=10                             # maximum time for chemsurf integration
 </CHEMSURF>
</SURFACE-MODEL>

<SOFC>
  op_pressure=101325.0               # operating pressure
  surfchem = y                        # surface reaction (y) or (n)
  open_circuit=n                     # open circuit condition (y) or (n)
  pot_step = 0.04                    # potential step for polarization curve
  <GEOMETRY model=button>             # opening tag ot specify the geometry of the cell geometry: button, planar or tubular
    nz=20                             # number of axial discretization points
    length=0.01                       # length of the cell (m)
    elperi = 1.2E-03                  # electrochemically active perimeter (m)
    <SECTION>
      name=air                        # fluid in the cathode channel : must be air
      height=1.0e-03                  # height of the channel
      width=1.0e-03                   # width of the channel
      ny=1                            # number of discretization in the y direction: must be 1
      <INLET>                         # opening tag to specify inlet conditions of the cathode stream
         u=5                         # inlet velocity of cathode stream (m/s)
         temperature = 1073.15        # inlet temperature of the cathode stream (K)
         pressure = 101325.0          # cathode channel pressure (Pa)
         <SPECIES>                    # cathode channel inlet species (Max 2 species)
           <MOLEFRAC>                 # mole fraction of the inlet species (Can also be <MASSFRAC>
              O2 0.21
              N2 *
           </MOLEFRAC>
```

```
        </SPECIES>                  # species closing tag for the cathode stream
      </INLET>                      # inlet condition closing tag for the cathode stream
    </SECTION>                      # close the cathode channel section
    <SECTION>                       # open the cathode boundary conditions and properties
      name=cathode                  # name of the section : must be cathode
      height=30.0d-06               # height of the cathode (m)
      width=1.0d-03                 # width of the cathode (m)
      ny=10                         # number of discretization in the y direction
      <POROUS model=DGM>            # porous media transport model (DGM or MIXED)
        porosity = 0.35             # porosity of the cathode
        tortuosity = 3.5            # tortuosity of the cathode
        pore_dia = 1.0E-06         # pore diameter of the cathode micro-structure (m)
        part_dia = 2.5e-06         # particle diameter of the cathode micro-structure (m)
        specific_area = 1080.0E02  # active specific area of the cathode (1/m)
      </POROUS>                     # close the porous media property tag
      <SPECIES>                     # species tag for cathode
        <MOLEFRAC>                  # initial guess for the cathode species (can also be <MASSFRAC>)
          O2  0.21
          N2  *
        </MOLEFRAC>                 # closing tag for the initial guess
      </SPECIES>                    # close the species tag
    </SECTION>                      # close the cathode section tag
    <SECTION>                       # open the section tag for electrolyte
      name=electrolyte              # name of the section : must be electrolyte
      height=2.5e-05                # height of the electrolyte (m)
      width=1.0e-03                 # width of the electrolyte (m)
      ny=1                          # number of discretization in the y direction
    </SECTION>                      # close the electrolyte tag
    <SECTION>                       # start the anode section tag
      name=anode                    # name of the section
      height=30E-06                 # height of the anode (m)
      width=1.0e-03                 # width of the anode (m)
      ny=15                         # number of dicretization in the y direction
      <POROUS model=DGM>
        porosity = 0.35             # porosity of the anode mictro-structure
        tortuosity = 3.5            # tortuosity of the anode micro-structure
        pore_dia = 1.0E-06         # pore diameter of the anode micro-structure (m)
        part_dia = 2.5e-06         # particle diameter of the anode micro-structure (m)
        specific_area = 1080.0E02  # active specific area of the anode (1/m)
      </POROUS>
      <SPECIES>                     # species tag for anode
        <MOLEFRAC>                  # initial guess for the anode species (can also be <MASSFRAC>)
          H2 0.5
          H2O *
        </MOLEFRAC>                 # closing tag for initial guess
      </SPECIES>                    # closing tag for the species
    </SECTION>                      # close the anode section
    <SECTION>                       # open the fuel channel section
      name=fuel                     # name of the section
      height=1.0e-03                # height of the fuel channel (m)
      width=1.0e-03                 # width of the fuel channel (m)
      ny=1                          # number of discretization in the y direction
      <INLET>                       # opening tag for inlet conditions for the fuel channel
        u=0.3                       # inlet velocity (m/s)
        temperature = 1073.15       # inlet temperature (K)
        pressure = 101325.0         # fuel channel pressure (Pa)
        <SPECIES>                   # opening tag for the fuel channel species
          <MOLEFRAC>                # mole fraction of the inlet species (can also be <MASSFRAC>)
            CH4   *
            H2O  0.5
          </MOLEFRAC>               # close the mole fraction tag
        </SPECIES>                  # closing tag for fuel channel species
      </INLET>                      # closing tag for fuel channel inlet conditions
    </SECTION>                      # close the fuel channel section
    </GEOMETRY>                     # closing tag for geometry specification
 <ELECTROCHEM>                      # opening tag to specify electrochemical properties
   Ecell = 0.7                      # operating cell potential
   Beta_a = 0.5                    # charge transfer coefficient for the anode charge transfer
   Beta_c = 0.5                    # charge transfer coefficient for the cathode charge transfer
   iH2 = 8.5                        # fit parameter for H2 oxidation exchange current density
   iO2 = 2.8                        # fit parameter for O2 reduction exchange current density
   kH2 =1.5E6                       # Pre exponential for H2 oxidation
   EH2 =100                         # activation energy for H2 oxidation (J/mol)
   kO2 =1.E6                        # pre exponential for O2 reduction
   EO2 =200                         # activation energy for O2 reduction (J/mol)
</ELECTROCHEM>                      # closing tag for electrochemical properties
<INIT>                              # general initial conditions
  p=101325.0                        # initial condition for pressure in the electrode(Pa)
  T=1073.15                         # initial temperature (K)
  isothermal=yes                    # this version supports only isothermal calculation !!!!!!!!!!!!!
</INIT>
<SOLVER>                            # opening tag to specify the solver set up parameters
  t_Begin = 0                      # initial integration time (s)
  t_End = 100                      # maximum integration time (s)
  dt = 1.0d-04                      # initial time step for the solver (s)
  <TOLERANCE>                       # tolerance for unknown parameters
   cont = 1.0d-05                   # tolerance for density
   u_Tol = 1.0d-05                 # tolerance for velocity
```

```
    s_Tol = 1.0d-04                # tolerance for species
    cdTol = 1.0d-05                 # tolerance for current density
   </TOLERANCE>                     # closing tag for tolerance specification
 </SOLVER>
 <OUTPUT>                           # opening tag to specify output
   species = mole                   # output will be in mole fractions
   file = 20                        # file number for the output file
   monitor =no                      # monitor option for output (y) or (n)
<OUTPUT>                            # closing tag for output
</SOFC>
```

The file ***sofc.inp*** holds all the parameters required to model a button or planar cell configuration. The most important tags and input parameters are separately described below.

<SOFC> tag

This is the opening tag for the sofc input parameters. The general behavior of the solution is controlled by the parameters specified right after this tag. They are described below

- **op_pre** this specifies the operating pressure.
- **surfchem** the surface chemistry can be stitched on or off by assigning *yes* or *no* to this option. The surface chemistry option may be switched off for a case of pure H_2 species, which saves the computation time.
- **open_circuit** when this option is specified as yes, then the electrochemistry solution is automatically switched off. In this case the fuel cell reactor behaves like a normal chemical reactor.
- **pot_step** potential step for polarization curve.

If one desires to do a button cell calculation then the pot_step option must be specified. If the button cell option is specified as *no* and the pot_step option is specified then the code will start simulating a polarization curve for a channel case. The code will start calculating the current for the cell potential specified by Ecell and then steps specified by pot_step will be taken. For eg. if Ecell is specified as 0.9 and pot_step as 0.1 then the code will start calculating the current for cell potentials 0.9, 0.8, 0.7 and so on until Ecell becomes 0.

<GEOMETRY model> tag

This tag is used to specify the geometrical configuration of the model. Four different configurations are possible as follows

- <GEOMETRY model=button>
- <GEOMETRY model=planar>
- <GEOMETRY model=tubular>
- <GEOMETRY model=planar_stack>

When *model* is specified as *button*, the code simulates a button cell configuration. A polarization curve cal also be generated for a button cell by specifying a value for the `pot_step`. The code can be used to simulate a planar or tubular cell configuration by specifying *planar* or *tubular* for the *model*. It should be kept in mind that the code simulates only axi-symmetric geometry for a tubular cell.While doing stack calculations the model must be specified as planar_stack.

<SECTION> tag

The various domains such as fuel channel, anode, electrolyte, cathode, and air channel are separated by the <SECTION> tag. Each section is given a name such as fuel, anode, electrolyte, cathode, and air. The <SECTION> tags for each domain holds sub-tags depending on the domain. For e.g. the electrodes contain sub-tags such as <porous>, <species> etc. Each of these sub-tags holds micro-structural properties and initial and boundary conditions relevant to that particular domain.

<MOLEFRAC> option for the anode and cathode

It is important to specify a right initial guess for the species mole or mass fractions in the anode and cathode. Generally air flows through the cathode channel. However, the user is free to specify pure oxygen running through the cathode channel by specifying the mass or mole fraction of O_2 as 1 in the air channel inlet conditions. In this case the most appropriate initial guess for the cathode species is 100% O_2. In simple terms please do not initialize the cathode with any species other than those specified in the air channel inlet.

It is quite necessary to have H_2O present as one of the species in the initial guess for the anode, irrespective of the species present in the fuel inlet condition. The code will fail under any circumstances, where the initial guess for the anode species do not contain H_2O.

<ELECTROCHEM> tag

It is quite necessary to clearly understand the meaning of various parameters specified in this tag for a physically realistic modeling of the cell.The following options are supported in <ELECTROCHEM> tag.

```
 <ELECTROCHEM>                  # opening tag to specify electrochemical properties
   Ecell = 0.7                  # operating cell potential
   Beta_a = 0.5                 # charge transfer coefficient for the anode charge transfer
   Beta_c = 0.5                 # charge transfer coefficient for the cathode charge transfer
   iH2 = 8.5                    # fit parameter for H2 oxidation exchange current density
   iO2 = 2.8                    # fit parameter for O2 reduction exchange current density
   kH2 =1.5E6                   # Pre exponential for H2 oxidation
   EH2 =100                     # activation energy for H2 oxidation (J/mol)
   kO2 =1.E6                    # pre exponential for O2 reduction
   EO2 =200                     # activation energy for O2 reduction (J/mol)
   constant=current             # calculation will be done by keeping the current constant
   avg_cd=0.3                   # current density in A/cm2 for which the calculation has to be done
</ELECTROCHEM>                  # closing tag for electrochemical properties
```

Ecell specifies the operating cell potential. In the case of button cell calculation or a polarization curve calculation, the code will start the calculation at the specified Ecell. The next potential the code chooses depends on the step size specified by the option *pot_step*. In short the code starts the calculation at the specified *Ecell* and then decreases the *Ecell* in steps specified by *pot_step* until 0 cell potential is reached. For example for H_2 button cell or channel calculation it is wise to specify *Ecell* as 1.2 V rather than starting the calculation at 0.8 V.

The parameters *Beta_a* and *Beta_c*, respective, are the charge transfer coefficients or asymmetry factor for anode and cathode half cell reactions. In generally their values are ~0.5. However, the user is free to choose the values.

The parameters *iH2* and *iO2* are the exchange current density parameters and are very important. For isothermal conditions, the exchange current density in DETCHEM^{SOFC} are modeled according to the following equations. The exchange current density for H_2 oxidation is given by

$$i_0 = i^*_{H_2} \frac{(p_{H_2}/p^*_{H_2})^{1/4}(p_{H_2O})^{3/4}}{1 + (p_{H_2}/p^*_{H_2})^{1/2}}, \tag{16.12}$$

and exchange current density for O_2 reduction is given by

$$i_0 = i^*_{O_2} \frac{(p_{O_2}/p^*_{O_2})^{1/4}}{1 + (p_{O_2}/p^*_{O_2})^{1/2}}. \tag{16.13}$$

iH2 and *iO2* stands for the parameters $i^*_{H_2}$ and $i^*_{O_2}$ respectively. For non-isothermal cases (even for isothermal) the parameters $i^*_{H_2}$ and $i^*_{O_2}$ can be further expressed in an Arrhenius form as shown below

$$i^*_{H_2} = k_{H_2} \exp\left(-\frac{E_{H_2}}{RT}\right) \tag{16.14}$$

$$i^*_{O_2} = k_{O_2} \exp\left(-\frac{E_{O_2}}{RT}\right). \tag{16.15}$$

In the input file parameters *kH2*, *kO2*, *EH2*, and *EO2* respectively stands for k_{H_2}, k_{O_2}, E_{H_2}, and E_{H_2} in the above equations. The user should either specify the parameters *iH2* and *iO2* or the parameters *kH2*, *kO2*, *EH2*, and *EO2*.

For more information on these topics the user is referred to the following references

1)H. Zhu, R. J. Kee, V. M. Janardhanan, O. Deutschmann, and D. G. Goodwin. Modeling Elementary Chemistry and Electrochemistry in Solid Oxide Fuel Cells, J. Electrochemical Society, (152) 12, A2427-A2440 (2005)

2)Vinod M. Janardhanan and Olaf Deutschmann. CFD analysis of a solid oxide fuel cell with internal reforming: Coupled interactions of transport, heterogeneous catalysis and electrochemical processes, J. Power Sources, (162) 2, 1192-1202(2006).

The electrochmical model equations can be solved by either fixing the operating cell potential or by fixing the current. The option **constant** can be used to specify the parameter the user wishes to keep as constant while solving the electrochemical model equations. The default value for **constant** is voltage. If the user whishes to keep cell current as constant then specify **constant**=current and specify the average current by using the input variable **avg_cd**.

<SOLVER> tag

check_negative option: Other than the input variables for the solver defined in the user input section, the check_negative option can also be used in case of convergence difficulties. In certain cases, if the code faces some convergence difficulties it is recommended to rerun the calculation with the option check_negative=yes. With this option switched on the calculations may turn slower. By default this option is switched off and in most cases the code runs successfully with this option switched off.

<TOLERANCE> tag

The *tolerance* tag holds the tolerance values for various dependent variables. In the case of convergence difficulties the user may relax the tolerance parameters. In all cases we recommend to try out the tolerance values specified in the examples.

<OUTPUT> tag

In the output tag the user can specify either mole fraction or mass fraction output for the species using the parameter *species*. When *mole* is specified species output will be in mole fraction and when *mass* is specified the species output will be in mass fractions. DETCHEM^{SOFC} generates a number of output file. The *file* option can be used to specify a file number that will be appended to each output file. Furthermore, a copy of the input file *sofc<file>.inp* will be generated. The option *monitor* can be used to monitor the solver integration. When *monitor* is specified as *no* the screen output will be for each axial position for a channel case and for each cell potential for a button cell case. A sample screen output for channel case is shown below.

```
CH4     H2O     H2      CO      CO2     O2      N2
NI(s)   H2O(s)  H(s)    OH(s)   CO(s)   C(s)    CH3(s)  CH2(s)  CH(s)
CH4(s)  O(s)    CO2(s)  HCO(s)
\n
*********************************************************
****                  DETCHEM SOFC                   ****
****    Vinod Janardhanan & Olaf Deutschmann         ****
****             Version 1.0 March 2006              ****
*********************************************************
\n
Solving cell potential   0.200000000000000
\n
 Axlpsn (m)  CD (A/cm2)  Erev(V)
 ---------------------------------
 0.0000E+00  5.4109E+00  8.3462E-01
 4.1667E-04  5.2078E+00  8.1632E-01
 1.2500E-03  4.9645E+00  7.9476E-01
 2.0833E-03  4.6781E+00  7.6961E-01
 2.9167E-03  4.3551E+00  7.4125E-01
 3.7500E-03  4.0113E+00  7.1082E-01
 4.5833E-03  3.6653E+00  6.7971E-01
 5.4167E-03  3.3311E+00  6.4898E-01
 6.2500E-03  3.0162E+00  6.1926E-01
 7.0833E-03  2.7239E+00  5.9088E-01
 7.9167E-03  2.4550E+00  5.6395E-01
```

In the case of button cell calculation the screen output contains only the cell potential and the corresponding current density as shown below

```
*********************************************************
****                   DETCHEM SOFC                  ****
****    Vinod Janardhanan & Olaf Deutschmann         ****
****              Version 1.0 March 2006             ****
*********************************************************
 1.1200E+00 -2.6356E-01
 1.1100E+00 -2.4497E-01
 1.1000E+00 -2.2329E-01
 1.0900E+00 -1.9790E-01
 1.0800E+00 -1.6803E-01
 1.0700E+00 -1.3275E-01
 1.0600E+00 -9.0957E-02
 1.0500E+00 -4.1322E-02
 1.0400E+00  9.5443E-03
 1.0300E+00  4.4429E-02
 1.0200E+00  8.1001E-02
 1.0100E+00  1.1919E-01
 1.0000E+00  1.5894E-01
 9.9000E-01  2.0017E-01
 9.8000E-01  2.4280E-01
```

16.4 Output

DETCHEM^{SOFC} generates a number of output files depending on whether the calculation is done for button cell or for planar cell. The various files, their contents and the cases for which these files are generated, are listed below

echem.plt

This file will be generated for button cell as well as for planar cell calculations. The name *echem* will be appended with the file number specified in the <OUTPUT> tag. For e.g. ***echem0010.plt***. This file contains the information regarding the electrochemical variables as a function of axial position. For a button cell calculation, the axial position will always be zero. The file contains variables in the following order

- z - Axia position (m) (0 for button cell calculation)
- Ecell - cell potential (V)
- Erev - Reversible potential or the Nernst potential (V)
- i - Current density (A/cm^2)
- eta_a - Anode activation over potential (V)
- eta_c - Cathode activation over potential (V)
- Rel - Ohmic resistance (Ω-m^2)
- Rct_a - Anode charge transfer resistance (Ω-cm^2)
- Rct_c - Cathode charge transfer resistance (Ω-cm^2)

The following figure 16.1 displays the current density as a function of axial position

fuel.plt

The file ***fuel.plt*** will be generated for planar cell calculations, and contains information related to the species and flow properties in the fuel channel. However, the user should keep in mind that, while calculating polarization curve for planar cell, the file ***fuel.plt*** will be overwritten for each operating cell potential. As in the previous case, the name *fuel* will be appended with the file number specified in the <OUTPUT> tag. For e.g. ***fuel0010.plt***. The file contains variables in the following order

- z - Axial position (m)
- rho - density (kg/m^3)
- u - velocity (m/s)
- Nre - Reynold number

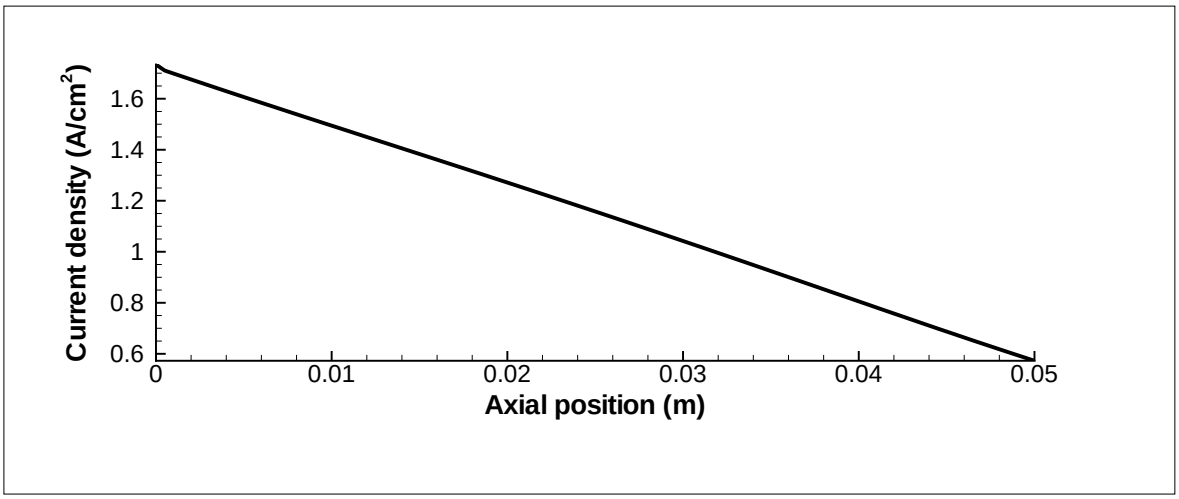


Figure 16.1: Current density as a function of axial position. The inlet fuel consists 66% H_2, 10%CH_4, 20% CO and rest H_2O. Operating cell potential 0.7 V, operating temperatyre 800^{o}C

- T - Temperature
- species (1 to K_g) - mole or mass fraction

The following figure 16.2 depicts an example of species profile within the fuel channel

air.plt

The file ***air.plt*** will be generated for planar cell calculations, and contains information related to the species and flow properties in the air channel. However, the user should keep in mind that, while calculating polarization curve for planar cell, the file ***air.plt*** will be overwritten for each operating cell potential. As in the previous case, the name *air* will be appended with the file number specified in the <OUTPUT> tag. For e.g. ***air0010.plt***. The file contains variables in the following order

- z - Axial position (m)
- rho - density (kg/m^3)
- u - velocity (m/s)
- Nre - Reynold number
- T - Temperature
- species (1 to K_g) - mole or mass fraction

outs.plt

The file ***outs.plt*** will be generated for planar cell calculations when surface chemistry is switched on and contains information related to the surface coverges of species in the anode. However, the user should keep in mind that, while calculating polarization curve for planar cell, the file ***outs.plt*** will be overwritten for each operating cell potential. As in the previous case, the name *outs* will be appended with the file number specified in the <OUTPUT> tag. For e.g. ***outs0010.plt***. The file contains variables in the following order

- z - Axial position (m)
- y - Radial position (m)
- species (1 to K_s) - surface coverages

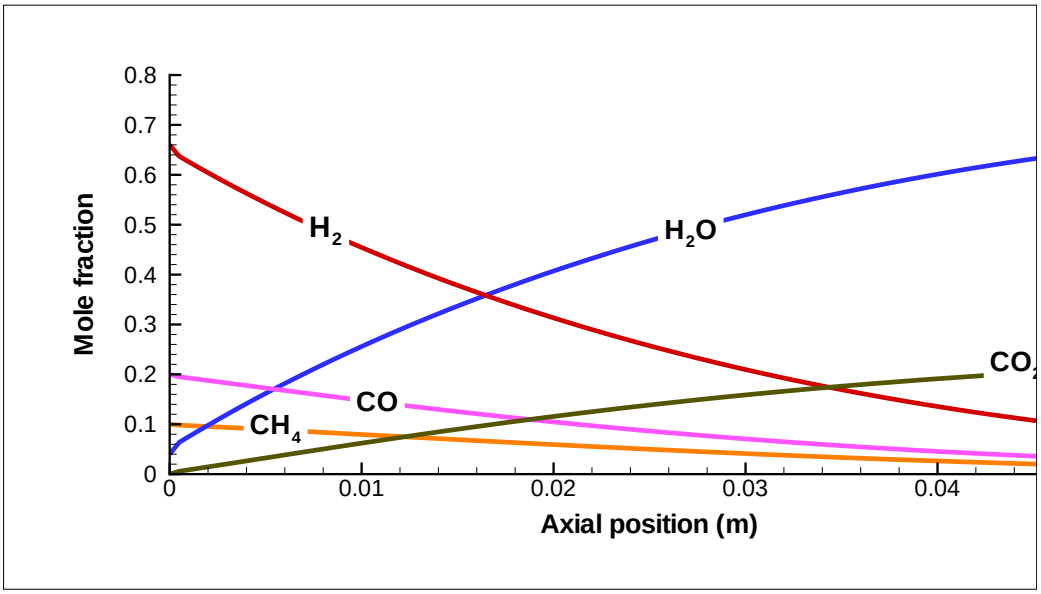


Figure 16.2: Species profile within the fuel channel as a function of axial position. The inlet fuel consists 66% H_2, 10%CH_4, 20% CO and rest H_2O. Operating cell potential 0.7 V, operating temperatyre 800^{o}C

summary.dat

The file ***summary.dat*** will be generated for each planar cell calculation and contains information on the average properties such as average current density, average activation losses, efficiency etc. If the file ***summary.dat***

exists, the the file will be appended with the information for every new calculation. The file contains information in the following order

- Ecell - Operating cell potential (V)
- Erev - Reversible cell potential (V)
- CD - Average current density (A/cm^2)
- PD - Average power density (W/cm^2)
- FU - Fuel utilization (%)
- Eff - Efficiency (%)
- AU - Air utilization (%)
- Act_A - Average anode activation loss (V)
- Act_C - Average cathode activation loss (V)
- R_el - Average ohmic resistance (Ω-m^2)
- Asp - Active specific area (1/m)
- La - Thickness of anode (m)
- Le - Thickness of electrolyte (m)
- Lc - Thickness of cathode (m)
- Ra_ct - Average charge transfer resistance of anode (Ω-cm^2)
- Rc_ct - Average charge transfer resistance of cathode (Ω-cm^2)

16.4.1 Examples

The examples directory with the distribution contains a sub directory named SOFC, which contains a number of case files. Other than the case files the directory also contains files such as ***thermdata***, ***moldata***, ***species.inp***, ***mech.inp***, and the surface chemistry mechanism file. The ***go*** file may be used to run the examples.

16.4.2 Setting up a problem

It is advised setup every new case in new directories. The directory should contains the following files; ***species.inp***, ***mech.inp***, ***sofc.inp***, ***thermdata***, ***moldata***, and the mechanism file specified in the ***mech.inp*** file.

16.4.3 Running the tool

After setting up the case directory the executable can be evoked by specifying the relative path or the absolute path to the binary. If the user wishes any name can be given for the input file ***sofc.inp***, and then use the batch script ***go*** supplied with the distribution to run the code. However, the user should make sure that the relative path present in the ***go*** correct according to his/her file structure.

Chapter 17

DC4OpenFOAM

17.1 Introduction

17.1.1 Modelling of Complex Flow with Detailed Chemistry

Modeling practical systems require the coupling of both arbitrary flow configurations including complex chemical reactions and additional conjugate heat transfer in solid bodies. From the point of view of simulation, those systems can be described with the following characteristics:

- three-dimensional laminar flow
- coupled heat transfer inside and between three-dimensional fluid and solid regions
- complex chemistry of gas-phase reactions
- complex chemistry of catalyzed surface reactions
- exchange of heat with the surroundings
- availability of temperature-dependent properties for all the compounds involved.

For the solution of problems of this type, there are, in principle, several CFD (Computational Fluid Dynamics) codes available (e.g., FLUENT STAR-CD, CFX). However, there are three constraints which restrict the simulation:

- availability of reliable, detailed gas-phase and surface mechanisms
- availability of computer power
- availability of finanical resources.

17.1.2 Simulation Tools DUO and DC4OpenFOAM

To reduce the costs of modeling, the calculation has to be fast and the software should to be powerful and cheap. The licensing of a commercial CFD code is very expensive and often not suitable for small companies. OpenFOAM is an open source CFD tool (free of cost) which enables the calculation of three-dimensional fluid and solid regions (www.openfoam.com). However, the handling of detailed thermophysical properties and chemical surface reactions is left to the user.

Parts of the DETCHEM software can be used in different ways additionally to OpenFOAM. The user has to distinct between two different approaches **DUO** and **DC4OpenFOAM**, which can be applied solely or together.

In **DUO** single tools of DETCHEM are called as external executables from OpenFOAM (*external coupling*). These tools calculate flow and the complex reactions in straight channels in one or two dimensions in space (axial or axial/radial). This feature can be used to calculate monolithic structures with many channels or tubes. In this case the flow regions inside the channels are *not* included in the CFD grid.

In **DC4OpenFOAM** some features of DETCHEM are provided to OpenFOAM calling a precompiled shared library (*internal coupling*). **DC4OpenFOAM** is a collection of tools from DETCHEM to handle thermophysical properties and detailed chemistry (gas phase and surface) within OpenFOAM. **DC4OpenFOAM** is an interface from DETCHEM to OpenFOAM. This capability is needed, if gas phase reactions in flow regions

outside of the channels of a monolith (e.g. upstream of a monolith) have to be considered. Additionally, the surface reactions at walls can be calculated. Therefore, it is also possible to calculate the full three-dimensional flow and the chemistry inside the channels of a monolith. In this case the flow regions inside the single channels would be included in the CFD grid.

Both tools, **DUO** and **DC4OpenFOAM** can be used independently or can be mixed up. The approach allows, for example, the calculation of a reformer in which the combined gas phase and surface chemistry is confined within geometrically simple tubes or channels and additional reactions take place in the gas phase out of the monolith and at the outer surface. In figure 17.1, a schematic drawing of a monolith with nine channels located in a surrounding flow field is shown. The flow covers the region upstream of the monolith as well as the channels in the monolith. Flow over the external surfaces of the catalyst tubes may be used to achieve thermal control. For example, hot exhaust products from an SOFC tail-gas combustor can be used to support endothermic steam reforming. Heat transfer occurs inside the fluid and the solid regions as well as at the interfaces between these regions.

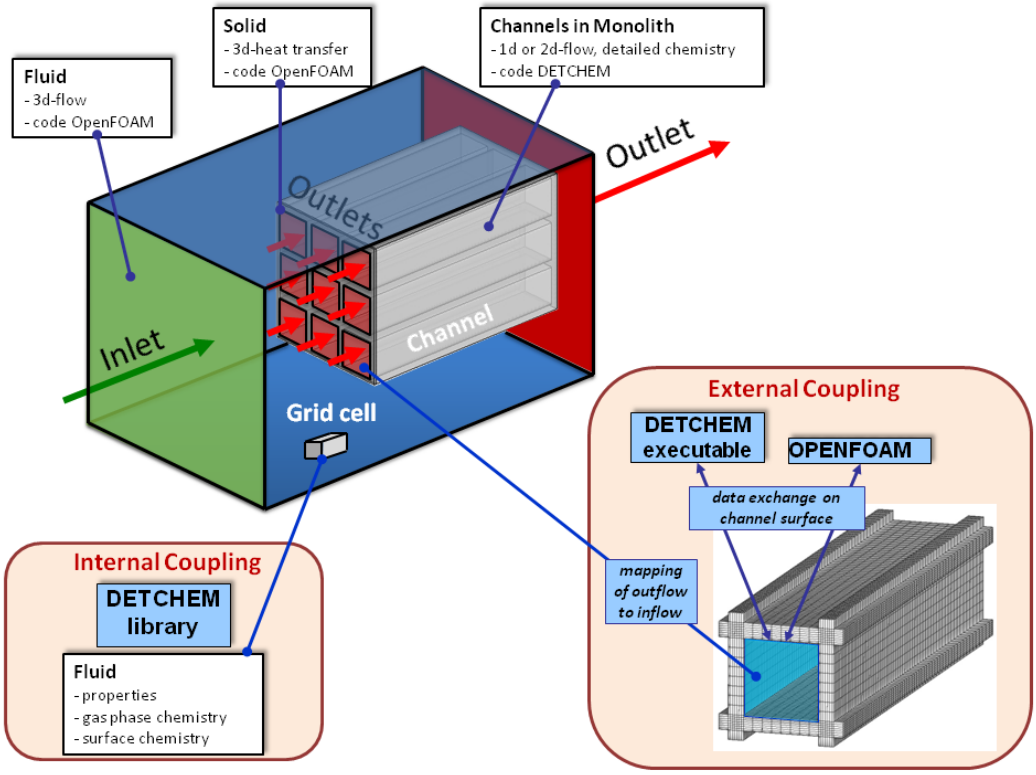


Figure 17.1: Typical configuration of a calculation domain

17.1.3 How DUO and DC4OpenFOAM work

A: DUO (External Coupling to OpenFOAM)

The capability of **DUO** is to model a reformer based on a monolithic structure. In figure 17.1, a schematic drawing of a monolith with nine channels located in a surrounding flow field is shown. The flow covers the region upstream of the monolith as well as the channels in the monolith. Heat transfer occurs inside the fluid and the solid regions as well as at the interfaces between these regions.

Using **DUO**, the flow outside of the monolith is calculated with the CFD code OpenFOAM, which handles essentially arbitrary geometry. In addition to the outer fluid flow, the code also solves the conjugate heat transfer in the solid structure of the monolith. This implies the handling of the heat transfer between the solid and fluid regions as well as the exchange of heat with the external environment. In the outer flow regions, the chemistry is often simple or can be fully neglected.

The flow inside the channels is calculated with DETCHEM codes by applying detailed gas-phase and surface chemistry. As the flow is handled as a one or two-dimensional flow, only a simple grid is used. This

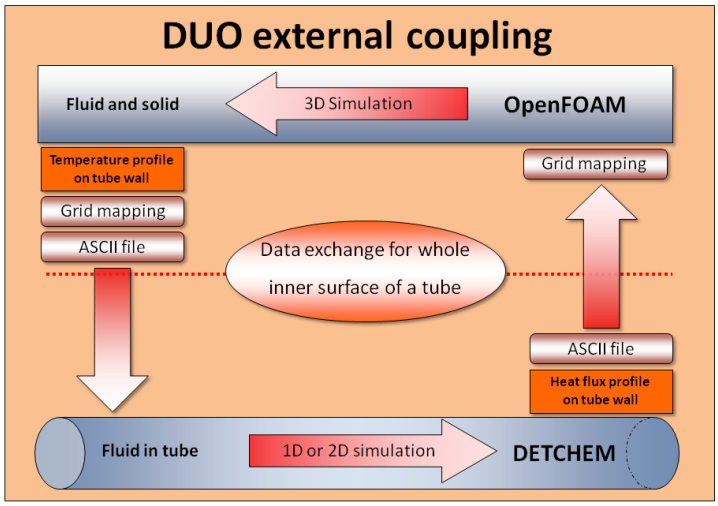


Figure 17.2: External Coupling between DETCHEM and OpenFOAM

makes the calculation fast. **DUO** handles the exchange of data at the inner surface of the channels between the two codes (see figure 17.2).

B: DC4OpenFOAM (Internal Coupling to OpenFOAM)

In the standard form OpenFOAM calculates the temperature dependent properties only via the Sutherland law. If the user wants to calculate a monolith with flow regions upstream or around a monolith, it might be insufficient to use this rough approach. Additionally chemical reactions may be of interest in these flow regions. There are solvers in OpenFOAM which can handle gas-phase reactions. But until now there is no capability to handle surface reactions. Within **DC4OpenFOAM** the calculation of properties and gas phase source terms is managed using a precompiled shared library which was made out of several parts of the DETCHEM toolbox. Figure 17.3 shows the principal structure of the *Internal Coupling*.

DC4OpenFOAM is a FORTRAN90 library which was made out of several parts of the DETCHEM software package. It implies the tools DETCHEM$^{\text{GASPROBE}}$ and DETCHEM$^{\text{SURFPROBE}}$ as well as the basic DETCHEM libraries.

At startup the library reads in:

- the thermodynamic database
- the kinetic theory parameters
- the gas-phase chemistry mechanism
- the surface chemistry reaction mechanism.

The library calculates some (not all!) properties of the fluid as well as the source terms of the gas phase and the surface kinetics for a fluid cell of the computational domain of a CFD-calculation with OpenFOAM. Therefore, the following cell variables are needed: pressure, gas/wall temperature, mass fractions of the gas phase species.

For each cell of the fluid domain calls to subroutines which are implied in the library are performed. For each cell the variables the pressure, the gas temperature and the concentrations of the gas phase species have to be provided to the library. From that, the properties heat conductivity, dynamic viscosity and the multi component diffusion coefficients of the gas phase species are send back to the CFD-code. The same holds for the calculation of the source terms of the gas phase kinetics. For the calculation of the surface reactions terms the temperature of the wall is needed additionally.

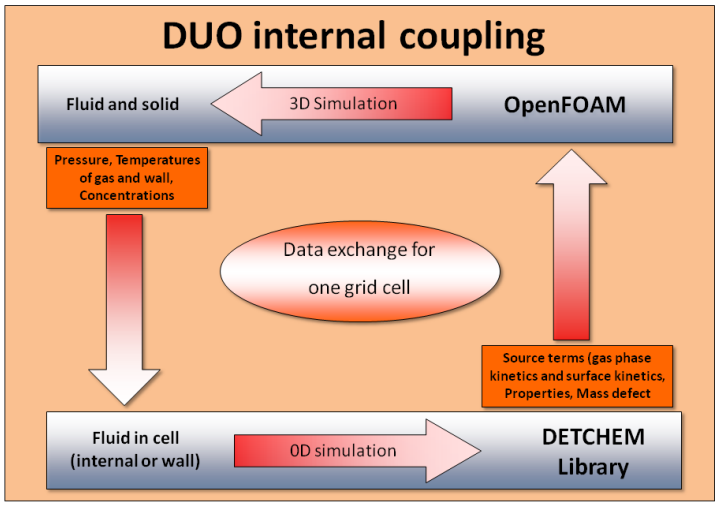


Figure 17.3: Internal Coupling between DETCHEM and OpenFOAM

C: Mixing of *External Coupling* and *Internal Coupling*

The two couplings can be mixed up. If we want, for example, to calculate a reformer, where the flow in the channels of the monolith should be calculated only in one or two dimensions the *external coupling* is sufficient. If we want also to consider the flow upstream of the monolith, we can do this within the range of the *external coupling* as long we dont̀ have to include chemical reactions in this regions and the determination of the temperature dependent properties based on the Sutherland law is enough. If we want to imply chemical reactions of the gas phase in the upstream region of a monolith and/or to determine the properties in more detail (could be important for mixture effects), we have to include additionally the *internal coupling*.

If we want to calculate the flow in the single channels of the monolith in three dimensions including gas phase and surface reactions we have to use the *internal coupling* solely.

17.2 Programming of DC4OpenFOAM

For the internal coupling of DETCHEM to OpenFOAM the library ***libDetchemFortran90-PropsAndSources.so*** was made out of several parts of the DETCHEM software package. Additionally some routines which are needed to make a new *solver* or *application* in OpenFOAM are provided. They are written in the language C++. For a deeper understanding of the code or information about the capabilities and the execution of OpenFOAM, the user should refer to the appropriate manual.

17.2.1 OpenFOAM

OpenFOAM (Open Field Operation and Manipulation) is a free, open source CFD software package developed by the OpenFOAM Team at SGI Corp and distributed by the OpenFOAM Foundation. It has a large user base across most areas of engineering and science, from both commercial and academic organisations. OpenFOAM has an extensive range of features to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to solid dynamics and electromagnetics. It includes tools for meshing and for pre- and post-processing. Almost everything (including meshing as well as pre- and post-processing) runs in parallel by default, enabling users to take full advantage of the computer hardware at their disposal.

By being open-source, OpenFOAM offers users complete freedom to customise and extend its existing functionality, either themselves or through support from OpenCFD. Code follows a highly modular design in which collections of functionality (e.g. numerical methods, meshing, physical models, etc.) are each compiled into their own shared library. Then, executable applications are created that are simply linked to the library functionality OpenFOAM, which includes over 80 solver applications that simulate specific problems in engineering mechanics and over 170 utility applications that perform pre- and post-processing tasks, e.g., meshing,

data visualisation, etc. For further information concerning the functionality and use of OpenFOAM, the user should refer to the manual.

17.2.2 Code Files

The new code files have to be copied into a new folder `$HOME/OpenFOAM/$USER-1.7.1/application/-chtMultiRegionSimpleFoamChemDuo`. The new solver ***chtMultiRegionSimpleFoamChemDuo*** has to be compiled. The internal coupling is managed using *two* libraries. The reader should carefully distinguish between these two libraries. As remarked before the shared FORTRAN90 library is called from **DUO** or the solver ***chtMultiRegionSimpleFoamChemDuo***, respectively. It is named ***libDetchemFortran90-PropsAndSources*** and can be found in the directory `<appDir>/libDetchemFortran90-PropsAndSources.so`. However, this library is *not* called directly from the solver or application ***chtMultiRegionSimpleFoamChemDuo***. It is called from an OpenFOAM library which is linked together with the solver as a shared object.

The OpenFOAM library is named ***libdetchemFoamModel.so*** and can be found in the directory `<appDir>/-detchemFoamModel`. As the library includes many routines, they are not explained in detail here. In Principle, this library is the gateway to the Fortran library and manages the transport models and thermophysical models.Both libraries have to be enlistet in the file `<caseDir>/system/`***controlDict***. This file allows the control of an OpenFOAM calculation. In case of *internal coupling*, the following entry has to be added:

```
libs
(
"libdetchemFoamModel.so"
"libDetchemFortran90-PropsAndSources.so"
);
```

17.3 Usage of DC4OpenFOAM

17.3.1 Installation and Test

A: Structure of Distribution

Figure 17.4 shows the structure of the folders of the distribution of **DC4OpenFOAM**.

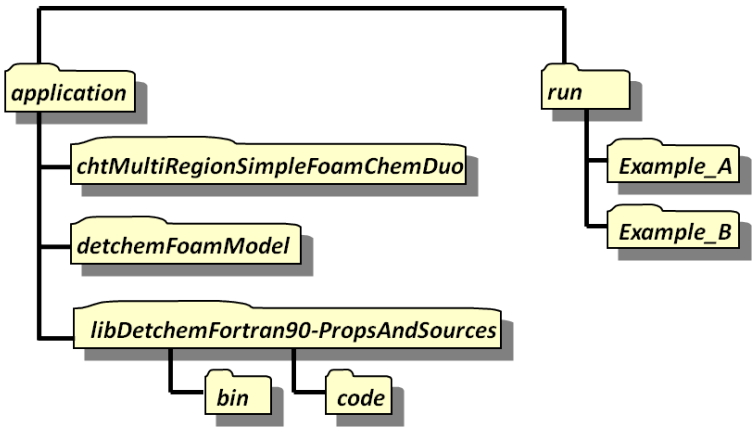


Figure 17.4: Folder Structure

The folders `application` and `run` have to be copied to the folder `$HOME/OpenFOAM/$USER-1.7.1` on your system.

B: Operating System

The installation was tested on two different PC systems A and B (see next table). In both cases version 1.7.1 of OpenFOAM was used. This version needs the compilers gcc-4.5 and g++-4.5. This led problems in configuration A, because the standard version of the compilers for this linux version is 4.6.2. For the installation and compiling of each code, the user should refer to the appropriate manuals. However, the installation including some basic commands which are necessary to compile the codes will be given in this chapter.

	System A	**System B**
Kernel Name	Linux 3.1.10-1.16-desktop	Linux 2.6.34-12-desktop
System software	openSUSE 12.1 (x86_64)	openSUSE 11.3 (x86_64)
OpenFOAM version	OpenFOAM1.7.1	OpenFOAM1.7.1
Compiler for OpenFOAM	gcc (SUSE Linux) 4.5.0 g++ (SUSE Linux) 4.5.0	gcc (SUSE Linux) 4.5.0 g++ (SUSE Linux) 4.5.0
Compiler for DETCHEM	gfortran	pgf90

C: Compiling of the individual Codes

To perform **DC4OpenFOAM**, it is necessary to install the following resources:

1. C++ resources:
   - CFD-Code OpenFOAM-1.7.1
   - OpenFOAM-1.7.1 solver ***chtMultiRegionSimpleFoamChemDuo***
   - OpenFOAM-1.7.1 library ***libDetchemFoamModel.so***
2. Fortran90 resources:
   - DETCHEM shared library ***libDetchemFortran90-PropsAndSources.so***

Compiling of library *libDetchemFortran90-PropsAndSources.so*

This library can be compiled in the directory `<appDir>/libDetchemFortran90-PropsAndSources` by typing the command ***make***. Without any options the command prints the capabilities of the makefile script. The user can compile single libraries from the different DETCHEM parts, maybe if something has to be changed. The libraries can also be packed together to the shared library to be used from **DUO**. The user can also compile the stand-alone executable ***duotest***, to check the functionality of the library. All compiled codes can be found in the folder `<appDir>/libDetchemFortran90-PropsAndSources/bin`. The name of the shared library to be used from **DUO** is ***libDetchemFortran90-PropsAndSources.so***. The compiled library has to be copied to the standard directory, where the libraries for OpenFOAM have to be present. This can be done with the command:

cp libDetchemFortran90-PropsAndSources ~/OpenFOAM/$USER-1.7.1/lib/linux64GccDPOpt

Compiling of Solver *chtMultiRegionSimpleFoamChemDuo*

The solver ***chtMultiRegionSimpleFoamChemDuo*** can be compiled in the directory `<appDir>/chtMultiRegionSimpleFoamChemDuo` by typing the command ***wmake***. If the code changes, it might be helpful to use the commands ***wmake clean*** and ***rmdepall*** firstly.

Compiling of Library *detchemFoamModel.so*

This library can be compiled in the directory `<appDir>/detchemFoamModel` by typing the command ***wmake***. If the code changes, it might be helpful to use the commands ***wmake clean*** and ***rmdepall*** firstly.

D: Test of the individual Codes

Before testing **DUO**, the user should ensure that the application ***chtMultiRegionSimpleFoamChemDuo***, and the library ***libDetchemFortran90-PropsAndSources.so*** work separately.

Test of *libDetchemFortran90-PropsAndSources*

The library can be tested in the directory `<appDir>/libDetchemFortran90-PropsAndSources/bin` by typing the command ***duotest***. ***duotest*** is an executable to run the shared library as a stand-alone tool. Be sure, that the keyword file ***properties.inp***, the data files ***moldata*** and ***thermdata*** and the mechanism files are present in a subdirectory named `detchem`. This is because in OpenFOAM the library is called in the `<caseDir>` and reads all information from the subdirectory `detchem`.

Test of OpenFOAM

After reading the instructions of the OpenFOAM manual the user can install and test the system by starting a test case of the standard installation.

Test of DUO

First of all one of the example cases should be tested. For the test of **DUO** for a new case a grid for a geometry consisting of a fluid part and a solid part has to be generated. All directories and files which are necessary for an OpenFOAM calculation have to be prepared. Firstly, the *application* should be tested without any coupling.

The internal name of the *application* is ***chtMultiRegionSimpleFoamChemDuo***. Each *application* is designed to be executed from a terminal command line, typically reading and writing a set of data files associated with a particular case. The command to start the OpenFOAM application in <caseDir> is (dont̀ forget the very last dot):

chtMultiRegionSimpleFoamChemDuo/case . .

To check the functionality of the application, all the requirements for the start of an OpenFOAM run have to be fulfilled. For testing, the coupling should be disabled in the file <caseDir>/***couplingParameters***:

`coupling off;` .

17.3.2 Preprocessing

A: Grid Generation in ICEM

OpenFOAM is able to use any mesh which was saved in the format needed for the code FLUENT. The grid can generated with GAMBIT, ICEM or any other software. In the actual section some important facts for the grid generation with the software ANSYS-ICEM are given.

1. Create two bodies named FLUID and SOLID for the two regions which shall be filled with flowing material and solid material. Normally, the body FLUID is inherently present. Add all blocks to the appropriate materials.

2. The interfaces for which surface chemistry will be applied have to be defined in the file `constant/FLUID/`***chemistryProperties*** after the keyword `surfacePatchName`. Example: `surfacePatchName FLUID_to_SOLID;`.

3. Convert the mesh to an unstructured mesh using the right mouse click on the entry *PreMesh* in the task bar.

4. Check the mesh carefully inside ICEM. No errors should arise.

5. On the strength of past experience it might happen, that an check in ICEM shows no errors whereas the check in OpenFOAM fails. If possible, firat load the mesh into FLUENT and perform an additional mesh check. If FLUENT accepts the mesh, OpenFOAM is also satisfied.

6. For writing out the grid the following parameters have to be used in the appropriate menu. Use for the item *Output Solver* the option *Fluent_V6* and for the item *Common Structural Solver* the option *ANSYS*. Be sure that the grid has the *unit [m]*. If not, scale the grid using a scaling factor in the menu for the output.

B: Grid Check in OpenFOAM

To use the grid (which was written in format for *FLUENT Version 6*) the data format has to be transformed, the regions have to be split and the mesh has to be checked. This can be done with the following steps:

1. **Generate case folder**

   Generate the folder <casDir>`/system`. Copy the following files into this folder (best from a previous calculation): ***controlDict***, ***fvSchemes***, ***fvSolution***.

2. **Transform the mesh**

   Perform the OpenFOAM application: ***fluent3DMeshToFoam ⟨grid file name⟩.msh -case***. Check carefully the output. If the code recognizes the two materials, some information should be printed which looks like:

```
Zone: 32 name: INTERFACE_SOLID_FLUID type: interface.  Reading zone data...done.
...
FINISHED LEXING
...
Creating faceZone 2 name: INTERFACE_SOLID_FLUID type: interface
```

3. **Split mesh**

   Perform the OpenFOAM application: ***splitMeshRegions -cellZones -overwrite***. Check carefully the output. If the code handles the two regions rightly, some information should be printed which look like this:

```
Region  Cells
------  -----
0       729
1       729
Region  Zone    Name
------  ----    ----
0       0       FLUID
1       1       SOLID
Sizes inbetween regions:
Region  Region  Faces
------  ------  -----
0       1       81
```

   Additionally you should find something like this:

```
For interface between region 0 and 1 added patch 8 FLUID_to_SOLID
```

4. **Check mesh**

   Perform OpenFOAM application: ***checkMesh -constant***. Check carefully the output. If something is wrong here, the calculation will not work.

C: Boundary Conditions in OpenFOAM

See example case for the two-zone solver ***chtMultiRegionSimpleFoam*** in the OpenFOAM manual.

D: Directory Structure of a Case (DUO and DC4OpenFOAM)

DUO and **DC4OpenFOAM** needs various objects (data files, keyword files, executables, libraries). The structure of a folder <caseDir> is shown in Fig. 3 1. There are many other objects needed for the calculation. In the folder only the most important files are shown. The objects in grey colour are needed for a calculation of a system with two regions. In this manual they are always named as FLUID and SOLID. The coloured objects have to be changed for the setup of a new case. They should be copied from one of the example cases and adapted afterwards. The objects, their functionality and content are explained in the following sections. The reader should always refer to figure 17.5 to get an overview about the data structure.

E: Data Input for OpenFOAM

Directories and files for the solver

The coupling of the codes OpenFOAM and DETCHEM is an *application* in OpenFOAM. The *application*, or *solver*, directories can be located anywhere but it is recommended they be within an application subdirectory of the users project directory, i.e., `$HOME/OpenFOAM/$USER-1.7.1/application`. In the following, this folder

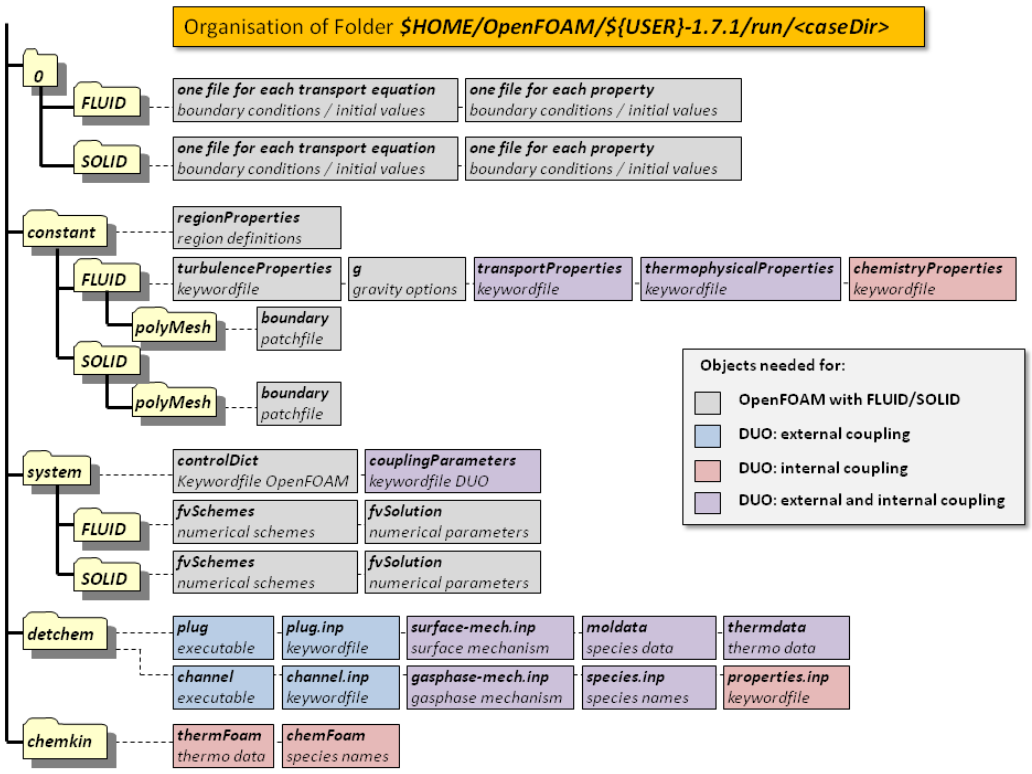


Figure 17.5: Folder structure of a calculation case

is called <appDir>. OpenFOAM *applications* are organised using a standard convention according to which the source code of each application is placed in a directory whose name is that of the *application*. The top-level source file takes the application name with the **.C** extension. The *application* name is ***chtMultiRegionSimpleFoamChemDuo***. Therefore, the folder where the application is located has the same name. The directory structure is as follows:

- <appDir>/chtMultiRegionSimpleFoamChemDuo
  - Make
  - fluid
  - solid
  - derivedFvPatchFields

The class and header files of the *application* **DC4OpenFOAM**, which are responsible for solving and *external* coupling, are directly located in the folder chtMultiRegionSimpleFoamChemDuo. The code files for the handling of the wall patches are located in the subdirectory derivedFvPatchFields. The C++ library which manages the external coupling to the Fortran90 library can be found in the directory: <appDir>/-detchemFoamModel.

Directories and files for the calculation

A calculation with OpenFOAM including all files is called a ***case***. Normally, a user would assign a name to a calculation ***case***, e.g. ***reformer***. This name becomes the name of a directory in which all the case files and subdirectories are stored. The case directories themselves can be located anywhere but it is recommended they be within a run subdirectory of the users project directory, i.e., $HOME/OpenFOAM/$USER-1.7.1. One advantage of this is that the $FOAM_RUN environment variable is set to $HOME/OpenFOAM/$USER-1.7.1/run by default; the user can quickly move to that directory by executing a preset alias ***run*** at the command line. In the following, the case folder is called <caseDir>.

As the coupling of the codes in **DC4OpenFOAM** is controlled via OpenFOAM, the appropriate folder structure for a calculation ***case*** also has to be applied. The basic directory structure for a ***case*** whose name is ***reformer*** is as follows:

- run/reformer
  - constant
  - system
  - <time>
  - detchem
  - chemkin

Directory	Content
constant	This directory contains a full description of the case mesh in a subdirectory polyMesh and files specifying physical properties for the application concerned, e.g. ***transportProperties***.
system	This directory is for setting parameters associated with the solution procedure itself. It contains at least the following files: ***controlDict***, in which run control parameters are set including start/end time, time step and parameters for data output; ***fvSchemes***, in which discretisation schemes used in the solution may be selected at run-time; and ***fvSolution***, in which the equation solvers, tolerances and other algorithm controls are set for the run. Additionally, the keyword file ***couplingParameters*** for control of **DUO** has to be located in this folder.
<time>	These directories contain individual files of data for particular fields. The data can be: either initial values and boundary conditions that the user must specify to define the problem or results written to file by OpenFOAM. Note that the OpenFOAM fields must always be initialised, even when the solution does not strictly require it, as in steady-state problems. The name of each time directory is based on the simulated time at which the data is written. Since we usually start our simulations at time t = 0, the initial conditions are usually stored in a directory named 0 or 0.000000e+00, depending on the name format specified. For example, the velocity field U and the pressure field p are initialised from files 0/FLUID/***U*** and 0/FLUID/***p***, respectively.
detchem	This directory contains all the files necessary for the *external coupling* of DETCHEM with **DUO**. This includes the executables named ***plug*** and/or ***channel*** as well as all other files required for a calculation, such as thermodynamic data, mechanisms, property file and the input files ***plug.inp*** and/or ***channel.inp***. Also, all result files from the DETCHEM calculations can be found in this folder.
chemkin	This directory contains the files necessary for the *internal coupling* of DETCHEM. The code has to know the species names and how many transport equations have to be solved. Also, the thermophysical data has to be present, because the properties heat capacities and enthalpy are calculated within OpenFOAM and not from the external coupling. Coming from a calculation with DETCHEM the user has to perform a three step approach to create the files needed.

1. Perform DETCHEM$^{\text{CHEMINP}}$ which is a pre-processor and translator for DETCHEM species and mechanism information files. DETCHEM$^{\text{CHEMINP}}$ requires one input file named ***cheminp.inp***. This file acts as a container for the species and mechanism information. If this information is stored in external files, e.g. species.inp and mech.inp, then ***cheminp.inp*** may simply look like the following example.

   ```
   <CHEMINP>
   {include species.inp}
   {include mech.inp}
   </CHEMINP>
   ```

   Like in all other DETCHEM applications, the files ***thermdata*** and ***moldata*** have to be present. The library DETCHEM$^{\text{CHEMINP}}$ produces several output files. The file ***chem.new*** contains the species and the mechanism information in a compact form. The files ***thermdata.new*** and ***moldata.new*** contain the thermodynamic data and the molecular properties (respectively) of the required species. The last file is not needed here.

2. Rename the files ***chem.new*** and ***thermdata.new*** to ***chem.inp*** and ***therm.dat***. Remove all information about mechanisms from the file ***chem.new***. Only the parts describing the elements and the species are needed.

3. Apply the OpenFOAM application ***converseChemkinToFoam*** to transform the files into a format which can be used form OpenFOAM. The output will be the two files ***chemFoam*** and ***thermFoam***. However, the user could also generate the files ***chem.inp*** and ***therm.dat*** using CHEMKIN files.

Important!

The user has to guarantee, that the count, the names and the order of the species in the file `<caseDir>/chemkin/`***chemFoam*** are the same as in the file `<caseDir>/detchem/`***properties.inp***. The species N_2 should be the last in each case.

F: Data Input for DETCHEM Library

The shared library ***libDetchemFortran90-PropsAndSources.so*** needs an input file ***properties.inp*** in the folder `<caseDir>/detchem`. A typical input file ***properties.inp*** with explanations is shown in the appendix. Additionally, the following data files are necessary:

thermdata	The thermodynamic database contains enthalpy, entropy and heat capacity data for each species.
moldata	This file contains the molar mass and kinetic theory parameters of chemical species.
gasphase-mech.inp	The gas-phase chemistry mechanism file lists the elementary reactions occurring in the gas phase. This file can be give any name, and this name should be mentioned in the ***mech.inp*** file.
surface-mech.inp	The surface chemistry reaction mechanism file ***surface-mech.inp*** lists the elementary reactions occurring on the surface. This file can be given any name and this name must be mentioned in the ***mech.inp*** file.
species.inp	For any DETCHEM application, the user needs to define the species that are to be considered. In many cases, it is advisable to put the declaration into a separate file, e.g. ***species.inp***.

Important!

The user has to guarantee, that the count, the names and the order of the species in the file ***properties.inp*** are the same as in the file `<caseDir>/chemkin/`***chemFoam***.

17.3.3 Control of DC4OpenFOAM

As usual, OpenFOAM is managed with the parameters in the ASCII-file ***controlDict*** which can be found in the subfolder `system` of the directory `<caseDir>`. In case of internal coupling, the properties and source terms are calculated from the shared Fortran90 library ***libDetchemFortran90-PropsAndSources.so***.

Fluid Model

The file `<caseDir>/constant/FLUID/`***turbulenceProperties*** has always to be present for the calculation. Use the file which can be found in the example cases. Nothing has to be changed.

Properties Model

The transport properties are managed in the file `<caseDir>/constant/FLUID/`***transportProperties***.

Thermophysical Model

The thermophysical models including the chemistry are managed in the file `<caseDir>/constant/FLUID/`***thermophysicalProperties***. In case of *external coupling* only, the file should include the following lines:

```
thermoType      hsPsiMixtureThermo<reactingMixture<gasThermoPhysics>>;
inertSpecie     N2;
```

In case of *internal coupling*, the following lines should be present:

```
thermoType      detchemHsPsiMixtureThermo<reactingMixture<gasThermoPhysics>>;
chemistryReader chemkinReader;
CHEMKINFile       \$FOAM_CASE/chemkin/chem.inp;
CHEMKINThermoFile \$FOAM_CASE/chemkin/therm.dat;
inertSpecie     N2;
```

Chemistry Model

The file <caseDir>/constant/FLUID/***chemistryProperties*** has always to be present if chemical reactions are calculated within the internal coupling. Use the file which can be found in the example cases.

17.3.4 Start of a Calculation with DC4OpenFOAM

Start the OpenFOAM application by typing the command ***chtMultiRegionSimpleFoamChemDuo***. All the keywords required for the coupling to DETCHEM (see table) have to be present in the file <caseDir>/-***couplingParameters***. Now, the entry for the keyword `coupling` has to be:

`coupling on;` .

For the first run, it is advisable to activate additional output by setting

`debugLevel 2;` .

You could also create an executable shell script named ***start*** with the following entry:

```
#!/bin/csh
chtMultiRegionSimpleFoamChemDuo
```

If you start the calculation with the command

```
Start > log &
```

the job will work in the background and the output will be written into the file ***log***.

17.3.5 Solution Data from OpenFOAM

A: Solution Data from OpenFOAM

The <time> directories contain the result files. For an example case with the name ***reformer***, the velocity field U for the timestep 10 s is stored in the file `run/reformer/10/`***U***. The solution files can be converted to the TECPLOT format using the utility ***foamToTecplot360***.

17.4 Appendix

17.4.1 Example of an Input File *properties.inp*

In the following, an example of a parameter file named ***properties.inp***, which is necessary to call the shared fortran90 library is printed. This file has to be located in the folder <caseDir>/detchem.

```
<SPECIES>
  <GASPHASE>
    NO  O2  NO2  O  N2
  </GASPHASE>

  <SURFACE mol/cm2=  2.72E-09>
  PT(s)
  O(s)
  NO2(s)
  NO(s)
  <INITIAL>
    PT(s)      *
    O(s)       0.0
    NO(s)      0.0
  </INITIAL>
  </SURFACE>

  <THIRD-BODY>
    MM = 0.4 O2 + 0.4 N2
  </THIRD-BODY>
</SPECIES>
```

```
<MECHANISM>
  <GASPHASE>
     file="detchem/noo2m"
  </GASPHASE>
  <SURFACE name="Pt">
    file="detchem/noo2sm"
  </SURFACE>
</MECHANISM>

<SURFACE-MODEL>
  <CHEMSURF>
    time=100
    analytical=no
    write_file=yes
  </CHEMSURF>
    Fcatgeo = 1
</SURFACE-MODEL>

<GASPROBE>
  T/K = 1200
  p/Pa= 100000
  <MOLEFRAC>
     NO2    0.1
     N2     *
  </MOLEFRAC>
</GASPROBE>
```

Chapter 18

DUO

18.1 Introduction

18.1.1 Background

The code **DUO** was developed to calculate the chemistry and the mass/heat transfer in monolithic reformers. The aim is to allow the user to optimise the design and the operating conditions of reformers based on numerical calculations.

At the entrance of a reformer, the flow is distributed to the channels. On one hand, the flow region upstream of the monolith is often geometrically complex but does not involve complex chemistry. Sometimes, the flow circulates around the monolith. On the other hand, the flow field in the single channels of the monolith is often simple (1D or 2D), but the chemistry is complex. An iterative coupling algorithm was developed that models the geometrically complex flow outside the monolith with one code and calculates the chemically complex reforming process in the channels of the monolith with another code. For this, the chemical models are designed to accommodate detailed gas-phase and catalytic reaction kinetics, possibly including hundreds of species and thousands of reactions. The developed software tool **DUO** controls the interaction between the two programmes.

18.1.2 Modelling of a Monolithic Reformer

Modeling practical systems requires the coupling of complex flow configurations with complex chemical-kinetics. From the point of view of simulation, reformers based on mini channels can be described with the following characteristics:

- three-dimensional laminar flow
- coupled heat transfer inside and between three-dimensional fluid and solid regions
- complex chemistry of gas-phase reactions
- complex chemistry of catalyzed surface reactions
- exchange of heat with the surroundings
- availability of temperature-dependent properties for all the compounds involved.

For the solution of problems of this type, there are, in principle, several CFD (Computational Fluid Dynamics) codes available (e. g., FLUENT STAR-CD, CFX). However, there are three constraints which restrict the simulation:

- availability of reliable, detailed gas-phase and surface mechanisms
- availability of computer power
- availability of finanical resources.

Reformers that are based on monolithic structures include thousands of equally sized small channels. To resolve these structures with numerical methods, grids with high spatial resolution are needed, which leads to large counts of cells. As the type of the basic differential equations is elliptical, the solution process is always iterative. The catalytic reforming kinetics of practical fuels (e.g., diesel) depend on gas-phase and heterogeneous reaction mechanisms, which may involve over a thousand elementary reactions. For each of the hundreds of species, a transport equation in a CFD code has to be solved. This would lead to simulation times in the time-span of weeks up to months or years, even if a large number of processors (e.g., several thousands) were available.

Therefore, direct coupling of chemistry at this level greatly exceeds the capability of computational fluid dynamics (CFD) models which can handle geometric complexity, but not in combination with complex chemistry. Therefore, the development and investigation of reformer systems using solely a common CFD code is not practical. Beyond that, the licensing of a commercial CFD code is very expensive and not suitable for small companies.

18.1.3 Simulation Tools DUO and DC4OpenFOAM

DUO stands for the coupling of the two computer codes DETCHEM Und (German for *and*) OpenFOAM and is a synonym for the joint utilisation of these two programmes. OpenFOAM is an open source CFD tool which enables the calculation of three-dimensional fluid and solid regions. It is free of cost. DETCHEM is a purchasable package of tools specifically designed for the modeling and simulation of reacting flows, particularly for heterogeneous systems such as catalysis, materials synthesis and fuel cells. The simulation tool **DUO** is the gateway between the two codes. The basic idea is to combine the advantages of the two programmes.

Parts of the DETCHEM software can be used in different ways additionally to OpenFOAM. The user has to distinct between two different approaches **DUO** and **DC4OpenFOAM**, which can be applied solely or together.

In **DUO** single tools of DETCHEM are called as external executables from OpenFOAM (*external coupling*). These tools calculate flow and the complex reactions in straight channels in one or two dimensions in space (axial or axial/radial). This feature can be used to calculate monolithic structures with many channels or tubes. In this case the flow regions inside the channels are *not* included in the CFD grid. The data exchange between the codes is managed with ASCII data files.

In **DC4OpenFOAM** some features of DETCHEM are provided by calling a precompiled shared library (*internal coupling*). This feature is used to calculate the fluid properties and the source terms of the chemical reactions in arbitrary regions. This capability is needed, if gas phase reactions in flow regions outside of the channels of a monolith (e. g. upstream of a monolith) have to be considered. Additionally, the surface reactions at walls can be calculated. Therefore, it is also possible to calculate the full three-dimensional flow and the chemistry inside the channels of a monolith. In this case the flow regions inside the single channels would be included in the CFD grid.

Both tools, **DUO** and **DC4OpenFOAM** can be used independently or can be mixed up. The approach allows, for example, the calculation of a reformer in which the combined gas phase and surface chemistry is confined within geometrically simple tubes or channels and additional reactions take place in the gas phase out of the monolith and at the outer surface. In figure 18.1, a schematic drawing of a monolith with nine channels located in a surrounding flow field is shown. The flow covers the region upstream of the monolith as well as the channels in the monolith. Flow over the external surfaces of the catalyst tubes may be used to achieve thermal control. For example, hot exhaust products from an SOFC tail-gas combustor can be used to support endothermic steam reforming. Heat transfer occurs inside the fluid and the solid regions as well as at the interfaces between these regions.

18.1.4 How DUO and DC4OpenFOAM work

A: DUO (External Coupling to OpenFOAM)

As mentioned above, the task is to model a reformer based on a monolithic structure. The approach exploits the structure of reformers in which the combined gas-phase and catalytic chemistry is confined within geometrically simple tubes or channels. In figure 18.1, a schematic drawing of a monolith with nine channels located in a surrounding flow field is shown. The flow covers the region upstream of the monolith as well as the channels in the monolith. Flow over the external surfaces of the catalyst tubes may be used to achieve thermal control. For example, hot exhaust products from an SOFC tail-gas combustor can be used to support endothermic steam reforming. Heat transfer occurs inside the fluid and the solid regions as well as at the interfaces between these regions.

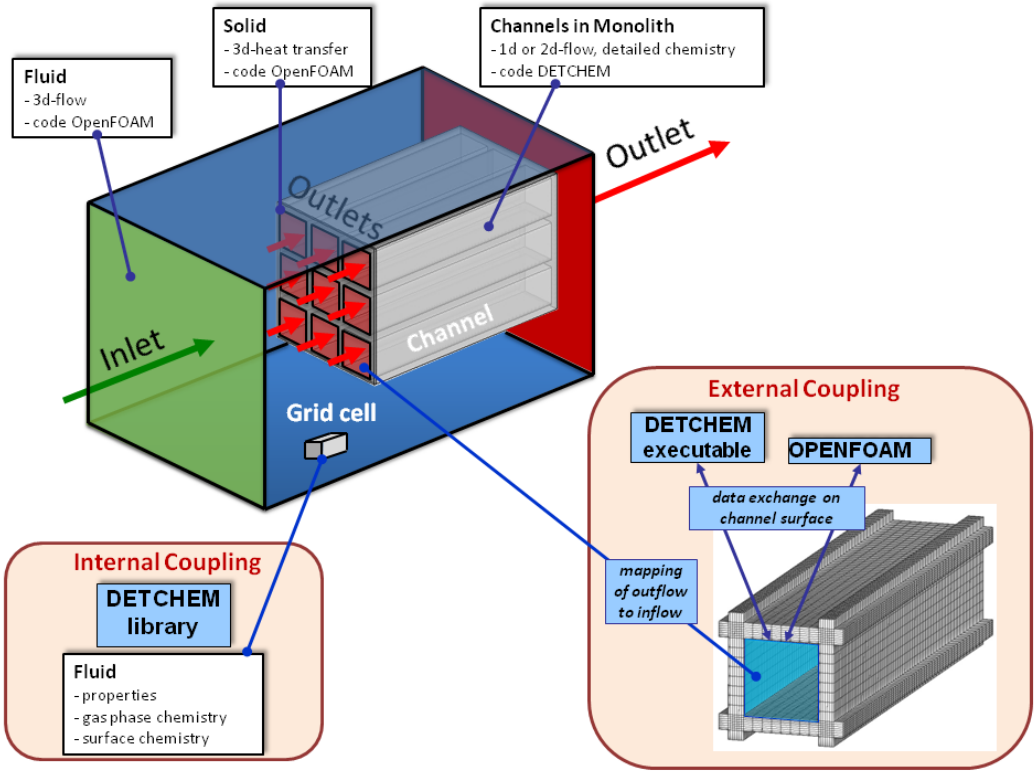


Figure 18.1: Typical configuration of a calculation domain

Using **DUO**, the flow outside of the monolith (see figure 18.1) is calculated with the CFD code OpenFOAM, which handles essentially arbitrary geometry. In addition to the outer fluid flow, the code also solves the conjugate heat transfer in the solid structure of the monolith. This implies the handling of the heat transfer between the solid and fluid regions as well as the exchange of heat with the external environment. In the outer flow regions, the chemistry is often simple or can be fully neglected.

The flow inside the channels is calculated with DETCHEM codes by applying detailed gas-phase and surface chemistry. As the flow is handled as a one or two-dimensional flow, only a simple grid is used. This makes the calculation fast.

DUO handles the exchange of data at the inner surface of the channels between the two codes (see figure 18.2). **DUO** provides the interface between the geometrically complex three-dimensional outer mesh and the one-dimensional axial mesh for the catalyst tubes. The internal chemistry problem is solved using axial wall temperature profiles that are specified by OpenFOAM. Tube wall heat fluxes are predicted as a result of the chemically reacting flow problem within the tubes. These flux profiles are supplied to OpenFOAM by DETCHEM. OpenFOAM then solves the external problem using the tube-wall heat fluxes as boundary conditions. The outer flow solution provides new tube-wall temperature profiles which are used to solve the chemistry problem. The iterative process continues until convergence between the inner and outer problems is achieved. The capability of **DUO** is quite general in the sense that there are essentially no restrictions on the external flow of geometry or the chemistry and transport within the tubes. Therefore **DUO** cannot only be used to model reformers, but also can be applied to simulate arbitrary systems where monolithic structures with internal chemistry are utilised.

DUO accomplishes two additional functions. First, the algorithm must interpolate between the three-dimensional mesh that is used by OpenFOAM and the one-dimensional band mesh used by DETCHEM. Second, **DUO** has to coordinate the iteration strategy between the outer CFD and the catalytic chemistry inside the tubes.

In addition to the fluid flow regions, the CFD mesh resolves all the solid regions. The CFD mesh contains cylindrical “cut-outs” corresponding to the inner walls of the tubes. In other words, the inner tube walls are seen as boundaries for the CFD problem. The cylindrical CFD face mesh on the inside of each tube wall is sectioned into a user-specified number of bands. The band spacing (one-dimensional mesh) is established to resolve gradients that characterize the interior catalytic chemistry problem. In the following discussion, “grid” refers to

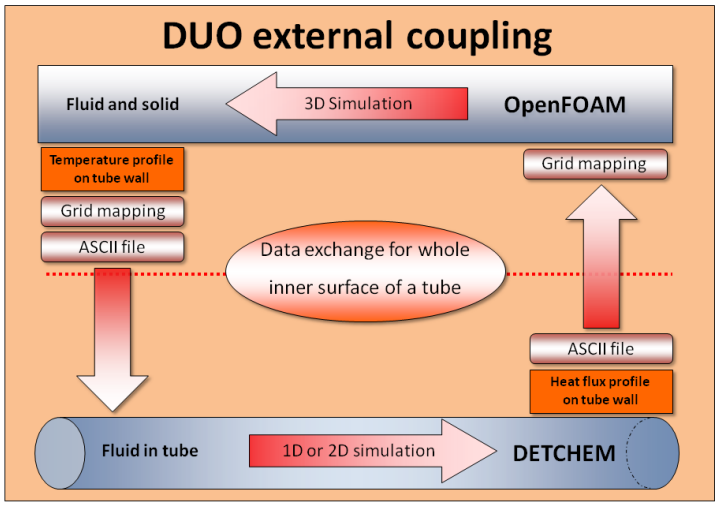


Figure 18.2: External Coupling between DETCHEM and OpenFOAM

the one-dimensional band discretization and "mesh" refers to the collection of faces in the CFD discretization of the tube wall. An integer band number is identified as the linearly increasing cell index for the grid, with the first band in the first grid cell being located at the tube inlet. **DUO** then loops over all mesh faces on a cylindrical tube and calculates the corresponding band number by projecting the face centroid onto the 1D grid.

Figure 18.3 illustrates the relationships between a general triangular surface mesh on a tube and the one-dimensional band grid. All triangular faces whose centroid touches the band are assigned to that band index. As can be seen, no special alignment is needed between the CFD mesh and the band grid. At the beginning of each solver iteration, **DUO** calculates face-averaged temperatures for all bands in a tube. These band averages correspond to azimuthally averaged temperatures on the tube wall faces in the band. **DUO** then creates a file with the list of the band temperatures and executes codes form DETCHEM. Typically, the chemistry model uses a spatial discretization which is different from the CFD grid. Therefore, interpolation of the temperature from the band grid to the chemistry grid is required.

If the user wishes, the conditions at the outflow areas of the CFD mesh, which cover the same areas as the inflows for the single channels, are also written to additional files and used for the calculation of the channels (s. Fig. figure 18.1). After that, **DUO** executes the code DETCHEM$^{\text{PLUG}}$ or DETCHEM$^{\text{CHANNEL}}$ for each channel. When the tube simulation is complete, the tool creates a file containing the heat flux profile for the band grid. **DUO** then analyses these heat-flux profiles and assigns the fluxes for each band to the corresponding collection of mesh faces in that band. This procedure is repeated for all tubes. The CFD code solves the outer flow and heat transfer problem with the prescribed heat fluxes as boundary conditions. The iterative process of averaging tube temperatures in bands, solving the catalytic chemistry problem, transferring the heat fluxes back to the CFD model, and solving the outer flow and temperature is repeated until convergence is achieved.

DUO can perform a user-specified number of CFD flow and thermal iterations between tube chemistry simulations. The inner tube chemistry calculation is typically run to convergence for each intermediate tube wall temperature profile as provided by the CFD model. In contrast, the CFD solution is always iterative, and hence, overall convergence is accelerated by performing several CFD iterations between running the chemistry model if the tube calculations are computationally expensive relative to the CFD calculations. It is possible that a reactor has a very large number of tubes such that modeling detailed chemistry in all the tubes is impractical. Therefore, the algorithm is generalized so that such groups of tubes can be modeled with one representative tube. The chemistry problem is solved for this representative tube, with the resulting heat fluxes being distributed back to all the tubes in the group.

B: DC4OpenFOAM (Internal Coupling to OpenFOAM)

In the standard form OpenFOAM calculates the temperature dependent properties only via the Sutherland law. If the user wants to calculate a monolith with flow regions upstream or around a monolith, it might be insufficient to use this rough approach. Additionally chemical reactions may be of interest in these flow regions. There are solvers in OpenFOAM which can handle gas-phase reactions. But until now there is no capability to handle

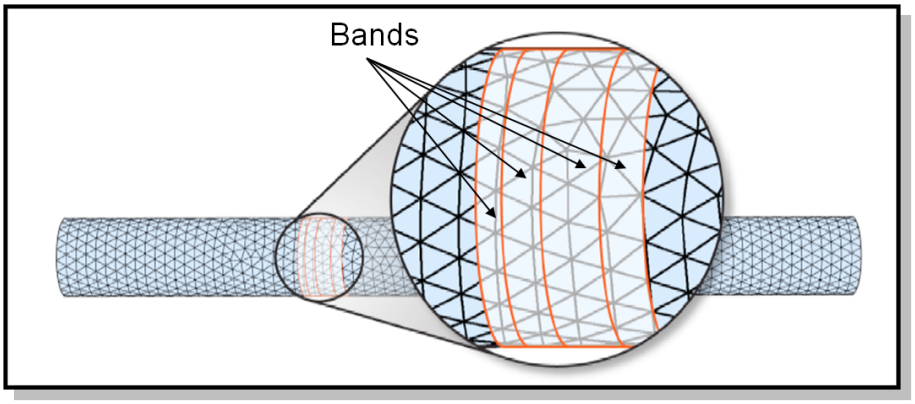


Figure 18.3: Mapping CFD mesh to 1D-grid

surface reactions. Within **DC4OpenFOAM** the calculation of properties and gas phase source terms is managed using a precompiled shared library which was made out of several parts of the DETCHEM toolbox. Figure 18.4 shows the principal structure of the *Internal Coupling*.

For each cell of the fluid domain calls to subroutines which are implied in the library are performed. For each cell the variables the pressure, the gas temperature and the concentrations of the gas phase species have to be provided to the library. From that, the properties heat conductivity, dynamic viscosity and the multi component diffusion coefficients of the gas phase species are send back to the CFD-code. The same holds for the calculation of the source terms of the gas phase kinetics. For the calculation of the surface reactions terms the temperature of the wall is needed additionally.

C: Mixing of External and Internal Coupling

The two couplings can be mixed up. If we want, for example, to calculate a reformer, where the flow in the channels of the monolith should be calculated only in one or two dimensions the *external coupling* (**DUO**) is sufficient. If we want also to consider the flow upstream of the monolith, we can do this within the range of the *external coupling* as long we don‘t have to include chemical reactions in this regions and the determination of the temperature dependent properties based on the Sutherland law is enough. If we want to imply chemical reactions of the gas phase in the upstream region of a monolith and/or to determine the properties in more detail (could be important for mixture effects), we have to include additionally the *internal coupling* **DC4OpenFOAM**. If we want to calculate the flow in the single channels of the monolith in three dimensions including gas phase and surface reactions we have to use the *internal coupling* **DC4OpenFOAM** solely.

18.2 Programming of DUO

In this chapter, some basic information about the programming is given. **DUO** is a collection of subroutines which allows to create a new *solver* or *application* in the world of OpenFOAM. For the external coupling of DETCHEM some changes were made in the original codes DETCHEM$^{\text{PLUG}}$ and DETCHEM$^{\text{CHANNEL}}$ to allow the exchange of data. The code is written in the language C++. The code lines are well commented. In general, the user should not change any code lines. For a deeper understanding of the code or information about the capabilities and the execution of OpenFOAM, the user should refer to the manual.

18.2.1 OpenFOAM

OpenFOAM (Open Field Operation and Manipulation) is a free, open source CFD software package developed by the OpenFOAM Team at SGI Corp and distributed by the OpenFOAM Foundation. It has a large user base across most areas of engineering and science, from both commercial and academic organisations. OpenFOAM has an extensive range of features to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to solid dynamics and electromagnetics. It includes tools for meshing and for pre- and post-processing. Almost everything (including meshing as well as pre- and post-processing) runs in parallel by default, enabling users to take full advantage of the computer hardware at their disposal.

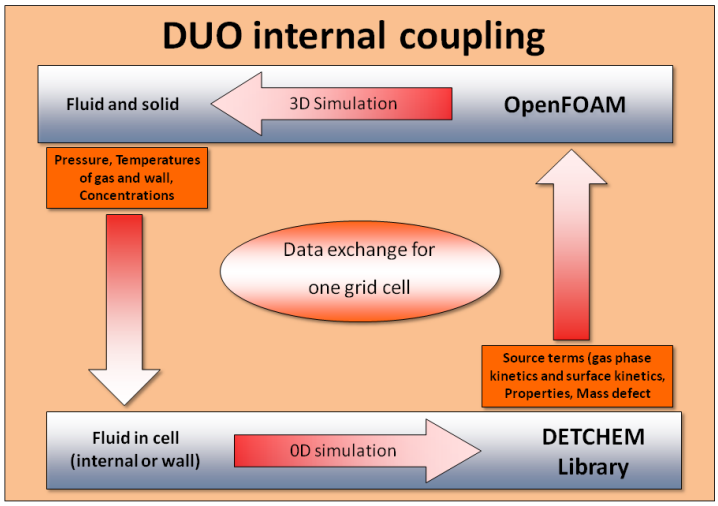


Figure 18.4: Internal Coupling between DETCHEM and OpenFOAM

By being open-source, OpenFOAM offers users complete freedom to customise and extend its existing functionality, either themselves or through support from OpenCFD. The code follows a highly modular design in which collections of functionality (e.g., numerical methods, meshing, physical models, etc.) are each compiled into their own shared library. Then, executable applications are created that are simply linked to the library functionality OpenFOAM, which includes over 80 solver applications that simulate specific problems in engineering mechanics and over 170 utility applications that perform pre- and post-processing tasks, e.g., meshing, data visualisation, etc.

For further information concerning the functionality and use of OpenFOAM, the user should refer to the appropriate manual.

DUO stands for the coupling of the two computer codes DETCHEM Und (German for *and*) OpenFOAM and is a synonym for the joint utilisation of these two programmes. Generally, we use the name **DUO** if we speak about the coupling of DETCHEM and OpenFOAM.

OpenFOAM is a C++ library used primarily to create executables, known as *applications*. OpenFOAM is distributed with a large set of precompiled applications but users also have the freedom to create their own or modify existing ones. The *applications* fall into two categories: *solvers*, that are each designed to solve a specific problem in continuum mechanics, and *utilities*, that are designed to perform simple pre-and post-processing tasks, mainly involving data manipulation and algebraic calculations. The OpenFOAM distribution contains numerous solvers and utilities covering a wide range of problems.

In this terminology, **DUO** is firstly a *solver*. Some coding was done by creating new subroutines which can be used to create a new *solver* or *application*. More precisely, **DUO** is a *solver* made of modules available in OpenFOAM. In the following, the terms *application* and *solver* are used synonymously. Additional code was written to handle the coupling. The resulting *solver* **DUO** calculates laminar flows with reactions in gas phase as well as surface reactions and conjugate heat transfer of connected fluid and solid regions. It calls DETCHEM$^{\mathrm{PLUG}}$ or DETCHEM$^{\mathrm{CHANNEL}}$ as external programs.

18.2.2 Code Files

The file containing the class definition takes a ***".C"*** extension, e.g., a class *nc* would be stored in the file ***nc.C***. As a means of checking errors, the piece of code being compiled must know that the classes it uses and the operations they perform actually exist. Therefore, each class requires a class declaration, contained in a header file with a ***".H"*** file extension, e.g., ***nc.H*** includes the names of the class and its functions. This file is included at the beginning of any piece of code using the class, including the class declaration code itself. Any piece of ***".C"*** code can resource any number of classes and must begin with all the ***".H"*** files required to declare these classes. The classes, in turn, can imply other classes and begin with the relevant ***".H"*** files. By searching recursively down the class hierarchy, we can produce a complete list of header files for all the classes on which the top level ***".C"*** code ultimately depends; these ***".H"*** files are known as the dependencies. With a dependency

list, a compiler can check whether the source files have been updated since their last compilation and selectively compile only those that need to be updated. Header files are included in the code using ♯ include statements, e.g., ♯ *include "otherHeader.H"* causes the compiler to suspend the reading from the current file and to read the file specified instead. Any self-contained piece of code can be put into a header file and included at the relevant location in the main code in order to improve code readability. The application directories can be located anywhere but it is recommended they be within an *application* subdirectory of the users project directory, i.e., `$HOME/OpenFOAM/$USER-1.7.1`. In the following, the folder where the applications can be found is called `<appDir>`. The code files which were additionally programmed in OpenFOAM to perform the external coupling to DETCHEM within the *application* ***chtMultiRegionSimpleFoamChemDuo*** are described shortly in the following table. They can be found in the directory `<appDir>/chtMultiRegionSimpleFoamChemDuo`.

Solver class files	
chtMultiRegionSimpleFoamChemDuo.C	Includes the OpenFOAM solver. Calls all subsequent routines of **DUO**.
Header files for coupling	
readCouplingParameters.H	Reading and interpretation of keyword file ***couplingParameters***.
initializeCoupling.H	Initialisation of coupling.
callDetchem.H	Calls DETCHEM$^{\text{PLUG}}$ or DETCHEM$^{\text{CHANNEL}}$ and copies input, output and exchange files.
writeOutletConditions.H	Calculates average conditions at outlets of CFD grid for all coupled tubes and writes to files. The data is used to set the inlet conditions for the DETCHEM calculations.
writeWallTemperatures.H	Calculates band-averaged wall temperatures for all coupled tubes and writes them to intermediate files.
Class and header files for boundary handling	
detchemHeatFluxTemperatureFvPatchScalarField.C *detchemHeatFluxTemperatureFvPatchScalarField.H*	Reads wall heat fluxes from intermediate files and maps them on the appropriate cells of the wall patches.

18.3 Usage of DUO

18.3.1 General

As **DUO** is the coupling of two codes, each code can be used as a stand-alone tool. For each program, several directories and files have to be present. For the use of each of the two programs, OpenFOAM and DETCHEM$^{\text{PLUG}}$ or DETCHEM$^{\text{CHANNEL}}$, the user should refer to the appropriate manuals. In this section, only the parts which are necessary for the control of the coupling between the two codes are described.

18.3.2 Installation and Test

A: Folder structure

Figure 18.5 shows the final structure of the folders.

The folders `application` and `run` have to be copied to the folder `$HOME/OpenFOAM/$USER-1.7.1` on your system. The folder `DETCHEM` can be copied to anywhere you want.

B: Operating System

The installation was tested on two different PC systems A and B (s Tab. 3 2). In both cases version 1.7.1 of OpenFOAM was used. This version needs the compilers gcc-4.5 and g++-4.5. This led problems in configuration A, because the standard version of the compilers for this linux version is 4.6.2. For the installation and compiling of each code, the user should refer to the appropriate manuals. However, the installation including some basic commands which are necessary to compile the codes will be given in this chapter.

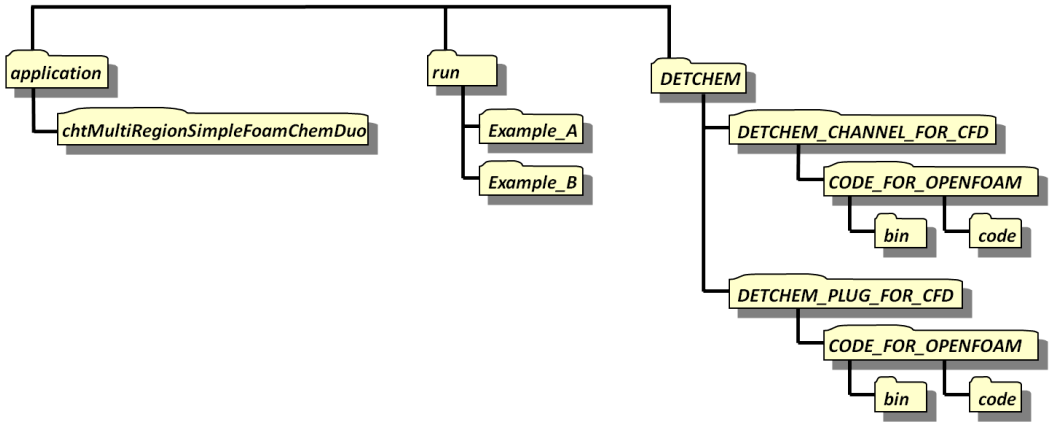


Figure 18.5: Folder Structure

	System A	**System B**
Kernel Name	Linux 3.1.10-1.16-desktop	Linux 2.6.34-12-desktop
System software	openSUSE 12.1 (x86_64)	openSUSE 11.3 (x86_64)
OpenFOAM version	OpenFOAM1.7.1	OpenFOAM1.7.1
Compiler for OpenFOAM	gcc (SUSE Linux) 4.5.0 g++ (SUSE Linux) 4.5.0	gcc (SUSE Linux) 4.5.0 g++ (SUSE Linux) 4.5.0
Compiler for DETCHEM	gfortran	pgf90

C: Compiling of the individual Codes

To perform **DUO**, it is necessary to install the following resources:

1. C++ resources:
   - CFD-Code OpenFOAM-1.7.1
   - OpenFOAM-1.7.1 solver ***chtMultiRegionSimpleFoamChemDuo***
2. Fortran90 resources:
   - DETCHEM executable ***plug***
   - DETCHEM executable ***channel***

Compiling of executable DETCHEM^{PLUG}

DETCHEM^{PLUG} can be compiled in the directory `<caseDir>/detchem/code` by typing the command ***make plug***. The executable ***plug*** is moved automatically to the folder which is one level higher `<caseDir>/detchem`.

Compiling of executable DETCHEM^{CHANNEL}

DETCHEM^{CHANNEL} can be compiled in the directory `<caseDir>/detchem/code` by typing the command ***make channel***. The executable ***channel*** is moved automatically to the folder which is one level higher `<caseDir>/detchem`.

Compiling of Solver *chtMultiRegionSimpleFoamChemDuo*

The solver ***chtMultiRegionSimpleFoamChemDuo*** can be compiled in the directory `<appDir>/chtMultiRegionSimpleFoamChemDuo` by typing the command ***wmake***. If the code changes, it might be helpful to use the commands ***wmake clean*** and ***rmdepall*** firstly.

D: Test of the individual Codes

Before testing **DUO**, the user should ensure that the application ***chtMultiRegionSimpleFoamChemDuo*** and the executables DETCHEM^{PLUG} and DETCHEM^{CHANNEL} work separately.

Test of DETCHEM^{PLUG}

DETCHEM^{PLUG} can be tested in the directory <caseDir>/detchem by typing the command ***plug***. ***plug*** is the executable which has to be compiled before running **DUO** in the directory <caseDir>/detchem/code. Running the code alone (without coupling) by using an input file ***plug.inp*** using all keywords, tags and includes necessary for a normal calculation allows to test the functionality.

Test of DETCHEM^{CHANNEL}

DETCHEM^{CHANNEL} can be tested in the directory <caseDir>/detchem by typing the command ***channel***. ***channel*** is the executable which has to be compiled before running **DUO** in the directory <caseDir>/detchem/code. Running the code alone (without coupling) by using an input file ***channel.inp*** using all keywords, tags and includes necessary for a normal calculation allows to test the functionality.

Test of OpenFOAM

After reading the instructions in the OpenFOAM manual the user can install and test the system by starting a test case of the standard installation.

Test of DUO

For the test of **DUO** for a new case a grid for a geometry consisting of a fluid part and a solid part has to be generated. All directories and files which are necessary for an OpenFOAM calculation have to be prepared. Firstly, the *application* **DUO** should be tested without any coupling. The internal name of the *application* **DUO** is ***chtMultiRegionSimpleFoamChemDuo***. Each *application* is designed to be executed from a terminal command line, typically reading and writing a set of data files associated with a particular case. The command to start the OpenFOAM application in <caseDir> is: ***chtMultiRegionSimpleFoamChemDuo -case .*** .

To check the functionality of the application, all the requirements for the start of an OpenFOAM run have to be fulfilled. For testing, the coupling has to be disabled in the file <caseDir>/couplingParameters:

```
coupling    off; .
```

18.3.3 Preprocessing

A: Grid Generation in ICEM

OpenFOAM is able to use any mesh which was saved in the format needed for the code FLUENT. The grid can generated with GAMBIT, ICEM or any other software. In the actual section some important facts for the grid generation with the software ANSYS-ICEM are given.

1. Create two bodies named FLUID and SOLID for the two regions which shall be filled with flowing material and solid material. Normally, the body FLUID is inherently present. Add all blocks to the appropriate materials.

2. Each interface for which an *external coupling* is wished in **DUO** has to be in an extra part. That means, that if a monolith has 100 channels you have to define 100 parts. Every interface between fluid and solid for which surface chemistry shall be calculated using the *internal coupling* has to be in one extra part.

3. Convert the mesh to an unstructured mesh using the right mouse click on the entry *PreMesh* in the task bar.

4. Check the mesh carefully inside ICEM. No errors should arise.

5. On the strength of past experience it might happen, that an check in ICEM shows no errors whereas the check in OpenFOAM fails. If possible, firat load the mesh into FLUENT and perform an additional mesh check. If FLUENT accepts the mesh, OpenFOAM is also satisfied.

6. For writing out the grid the following parameters have to be used in the appropriate menu. Use for the item *Output Solver* the option *Fluent_V6* and for the item *Common Structural Solver* the option *ANSYS*. Be sure that the grid has the unit [m]. If not, scale the grid using a scaling factor in the menu for the output.

B: Grid Check in OpenFOAM

To use the grid (which was written in format for *FLUENT Version 6*) the data format has to be transformed, the regions have to be split and the mesh has to be checked. This can be done with the following steps:

1. **Generate case folder**

   Generate the folder <casDir>/system. Copy the following files into this folder (best from a previous calculation): ***controlDict***, ***fvSchemes***, ***fvSolution***.

2. **Transform the mesh**

   Perform the OpenFOAM application: ***fluent3DMeshToFoam ⟨grid file name⟩.msh -case***. Check carefully the output. If the code recognizes the two materials, some information should be printed which looks like:

   ```
   Zone: 32 name: INTERFACE_SOLID_FLUID type: interface. Reading zone data...done.
   ...
   FINISHED LEXING
   ...
   Creating faceZone 2 name: INTERFACE_SOLID_FLUID type: interface
   ```

3. **Split mesh**

   Perform the OpenFOAM application: ***splitMeshRegions -cellZones -overwrite***. Check carefully the output. If the code handles the two regions rightly, some information should be printed which look like this:

   ```
   Region  Cells
   ------  -----
   0       729
   1       729
   Region  Zone    Name
   ------  ----    ----
   0       0       FLUID
   1       1       SOLID
   Sizes inbetween regions:
   Region  Region  Faces
   ------  ------  -----
   0       1       81
   ```

   Additionally you should find something like this:

   ```
   For interface between region 0 and 1 added patch 8 FLUID_to_SOLID
   ```

4. **Check mesh**

   Perform OpenFOAM application: ***checkMesh -constant***. Check carefully the output. If something is wrong here, the calculation will not work.

C: Boundary Conditions in OpenFOAM

For the "normal" boundary conditions see the example case for the two-zone solver ***chtMultiRegionSimpleFoam***. The "special" boundary conditions, which are necessary to manage the coupling are described later.

D: Directory Structure of a Case (DUO and DC4OpenFOAM)

DUO and **DC4OpenFOAM** need various objects (data files, keyword files, executables, libraries). The structure of a folder <caseDir> is shown in figure 18.6. There are many other objects needed for the calculation. In the folder only the most important files are shown. The objects in grey colour are needed for a calculation of a system with two regions. In this manual they are always named as FLUID and SOLID. The coloured objects have to be changed for the setup of a new case. They should be copied from one of the example cases and adapted afterwards. The objects, their functionality and content are explained in the following sections. The reader should always refer to figure 18.6 to get an overview about the data structure.

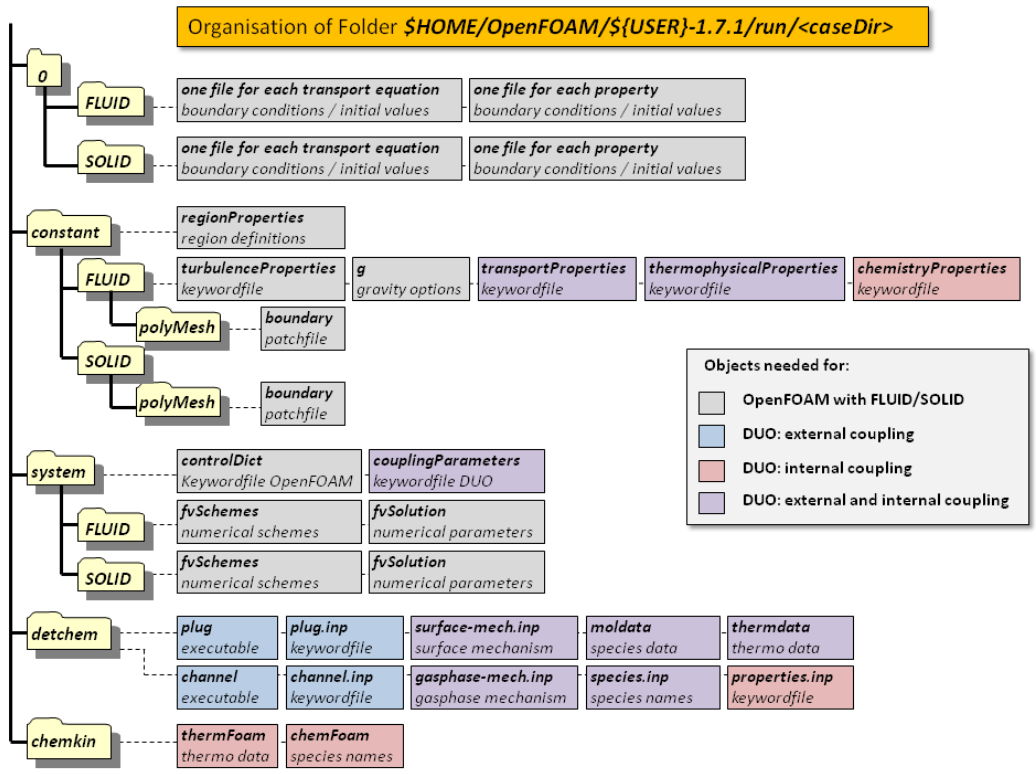


Figure 18.6: Folder structure of a calculation case

E: Data Input for OpenFOAM

Directories and files for the solver The coupling of the codes OpenFOAM and DETCHEM is an *application* in OpenFOAM. The *application*, or *solver*, directories can be located anywhere but it is recommended they be within an application subdirectory of the users project directory, i.e., `$HOME/OpenFOAM/$USER-1.7.1/-application`. In the following, this folder is called <appDir>. OpenFOAM *applications* are organised using a standard convention according to which the source code of each application is placed in a directory whose name is that of the *application*. The top-level source file takes the application name with the ***".C"*** extension. The *application* name is ***chtMultiRegionSimpleFoamChemDuo***. Therefore, the folder where the application is located has the same name. The directory structure is as follows:

- <appDir>/chtMultiRegionSimpleFoamChemDuo
  - Make
  - fluid
  - solid
  - derivedFvPatchFields

The class and header files of the *application* **DUO**, which are responsible for solving and *external* coupling, are directly located in the folder dirchtMultiRegionSimpleFoamChemDuo. The code files for the handling of the wall patches are located in the subdirectory dirderivedFvPatchFields.

Directories and files for the calculation

A calculation with OpenFOAM including all files is called a ***case***. Normally, a user would assign a name to a calculation ***case***, e.g., ***reformer***. This name becomes the name of a directory in which all the case files and subdirectories are stored. The case directories themselves can be located anywhere but it is recommended they be within a `run` subdirectory of the users project directory, i.e., *HOME*/*OpenFOAM*/`USER-1.7.1.`One advantage of this is that the `$FOAM_RUN` environment variable is set to `$HOME/OpenFOAM/$USER-1.7.1/run` by default; the user can quickly move to that directory by executing a preset alias, ***run***, at the command line.

In the following, the case folder is called <caseDir>. As the coupling of the codes in **DUO** is controlled via OpenFOAM, the appropriate folder structure for a calculation case also has to be applied. The basic directory structure for a *case* whose name is *reformer* is as follows:

- run/reformer
  - constant
  - system
  - <time>
  - detchem

Directory	Content
constant	This directory contains a full description of the case mesh in a subdirectory polyMesh and files specifying physical properties for the application concerned, e.g. ***transportProperties***.
system	This directory is for setting parameters associated with the solution procedure itself. It contains at least the following files: ***controlDict***, in which run control parameters are set including start/end time, time step and parameters for data output; ***fvSchemes***, in which discretisation schemes used in the solution may be selected at run-time; and ***fvSolution***, in which the equation solvers, tolerances and other algorithm controls are set for the run. Additionally, the keyword file ***couplingParameters*** for control of **DUO** has to be located in this folder.
<time>	These directories contain individual files of data for particular fields. The data can be: either initial values and boundary conditions that the user must specify to define the problem or results written to file by OpenFOAM. Note that the OpenFOAM fields must always be initialised, even when the solution does not strictly require it, as in steady-state problems. The name of each time directory is based on the simulated time at which the data is written. Since we usually start our simulations at time t = 0, the initial conditions are usually stored in a directory named 0 or 0.000000e+00, depending on the name format specified. For example, the velocity field U and the pressure field p are initialised from files 0/FLUID/***U*** and 0/FLUID/***p***, respectively.
detchem	This directory contains all the files necessary for the *external* coupling of DETCHEM. This includes the executables named ***plug*** and/or ***channel*** as well as all other files required for a calculation, such as thermodynamic data, mechanisms, property file and the input files ***plug.inp*** and/or ***channel.inp***. Also, all result files from the DETCHEM calculations can be found in this folder.

1. Perform DETCHEM$^{\text{CHEMINP}}$ which is a pre-processor and translator for DETCHEM species and mechanism information files. DETCHEM$^{\text{CHEMINP}}$ requires one input file named ***cheminp.inp***. This file acts as a container for the species and mechanism information. If this information is stored in external files, e.g. species.inp and mech.inp, then ***cheminp.inp*** may simply look like the following example.

   ```
   <CHEMINP>
   {include species.inp}
   {include mech.inp}
   </CHEMINP>
   ```

   Like in all other DETCHEM applications, the files ***thermdata*** and ***moldata*** have to be present. The library DETCHEM$^{\text{CHEMINP}}$ produces several output files. The file ***chem.new*** contains the species and the mechanism information in a compact form. The files ***thermdata.new*** and ***moldata.new*** contain the thermodynamic data and the molecular properties (respectively) of the required species. The last file is not needed here.

2. Rename the files ***chem.new*** and ***thermdata.new*** to ***chem.inp*** and ***therm.dat***. Remove all information about mechanisms from the file ***chem.new***. Only the parts describing the elements and the species are needed.

3. Apply the OpenFOAM application ***converseChemkinToFoam*** to transform the files into a format which can be used form OpenFOAM. The output will be the two files ***chemFoam*** and ***thermFoam***. However, the user could also generate the files ***chem.inp*** and ***therm.dat*** using CHEMKIN files.

Important!

The user has to guarantee, that the count, the names and the order of the species in the file <caseDir>/chemkin/***chemFoam*** are the same as in the file <caseDir>/detchem/***properties.inp***. The species N_2 should be the last in each case.

F: Data Input for DETCHEM

Before running ***plug*** (which is the executable of DETCHEM^{PLUG}), the user must prepare an input file ***plug.inp*** in the folder <caseDir>/detchem. A typical input file ***plug.inp*** with explanations is shown in the appendix. Before running ***channel*** (which is the executable of DETCHEM^{CHANNEL}, the user must prepare an input file ***channel.inp*** in the folder <caseDir>/detchem. A typical input file ***channel.inp*** with explanations is shown in the appendix. As the input file is the same for all calculated tubes, the file contains so-called *includes*. These tags allow the substitution of values in the input file with values written into intermediate files by **DUO**. Additionally, the following data files are necessary:

thermdata	The thermodynamic database contains enthalpy, entropy and heat capacity data for each species.
moldata	This file contains the molar mass and kinetic theory parameters of chemical species.
gasphase-mech.inp	The gas-phase chemistry mechanism file lists the elementary reactions occurring in the gas phase. This file can be give any name, and this name should be mentioned in the ***mech.inp*** file.
surface-mech.inp	The surface chemistry reaction mechanism file ***surface-mech.inp*** lists the elementary reactions occurring on the surface. This file can be given any name and this name must be mentioned in the ***mech.inp*** file.
species.inp	For any DETCHEM application, the user needs to define the species that are to be considered. In many cases, it is advisable to put the declaration into a separate file, e.g. ***species.inp***.

18.3.4 Control of DUO

As usual, OpenFOAM is managed with the parameters in the ASCII-file ***controlDict*** which can be found in the folder <caseDir>/system. In case of *external coupling* OpenFOAM calls the executables ***plug*** or ***channel*** from DETCHEM. The coupling between the two codes is managed using the ASCII keyword list ***couplingParameters*** which is located in the same directory. Among others, two special types of keywords are used: *dictionaries* and *lists*.

Special boundary conditions for coupling

DUO handles the exchange of data at the inner surface of the channels between the two codes (see figure 18.2). In addition to the fluid flow regions, the CFD mesh resolves all the solid regions. The CFD mesh contains cylindrical "cut-outs" corresponding to the inner walls of the tubes. In other words, the inner tube walls are these boundaries for the CFD problem where data with DETCHEM is interchanged. To handle the data transfer new boundary conditions for OpenFOAM have been defined. They can be found in the directory <appdir>/chtMultiRegionSimpleFoamChemDuo/derivedFvPatchFields/ detchemHeatFluxTemperature. Every tube interchange surface has to be defined with the boundary condition name detchemHeatFluxTemperature in the file <time>/SOLID/***T***. Here an example for the boundary WALL_0101. The values for the parameters K, q, *gradient* and *value* are arbitrary, because they are not used.

```
WALL_0101
    {
        type            detchemHeatFluxTemperature;
        K               K;
        q               uniform 0;
        gradient        uniform 0;
        value           uniform 0;
    }
```

Dictionaries

OpenFOAM uses *dictionaries* as the most common means of specifying data. A *dictionary* is an entity that contains a set of data entries. The keyword entries follow the general format:

```
<keyword>  <dataEntry1> ... <dataEntryN>;}
```

Most entries are single data entries of the form of:

```
<keyword> <dataEntry1>;
```

Most OpenFOAM data files are dictionaries containing a set of keyword entries. Dictionaries provide the means for organising entries into logical categories and can be specified hierarchically so that any dictionary can contain one or more *subdictionary* entries. The format for a dictionary is to specify the dictionary name followed by the entries enclosed in curly braces "{}" as follows:

```
<dictionaryName>
{
... keyword entries ...
};
```

List Structures

A second keyword structure is a *list*. The format of a *list* comprises the specification of the list name followed by the entries enclosed in round parentheses "()" as follows:

```
<listName>
(
... entries ...
);
```

Keywords for DUO

The next table shows all keywords and data elements in the file ***couplingParameters*** required by **DUO**. The column **type** describes the value of the `<dataEntry>`. Entries of the type *R* demand a real value after the keyword name, and those of the type *I* demand an integer value. Keywords of the type *C* demand a character string. The type *Dict* stands for a *dictionary*. The type *List* contains multiple elements of the same type (each of type *C* or *I*). In the following, the term *patch* is used to define a boundary condition. For the purpose of applying boundary conditions, a boundary is generally broken up into a set of *patches*. One *patch* may include one or more enclosed areas of the boundary surface which do not necessarily need to be physically connected.

Caution: All keywords have to be included in the file. There are no default values. The dictionaries and lists also have to be present. If they are not used, they have to be empty (e.g. `namesOfCoupledOutlets( );`).

Keyword structure	Type	Unit	Meaning	Values
General				
`coupling`	*C*	-	Switches the external coupling to DETCHEM and controls which DETCHEM code will be coupled.	plug, channel, off
`debugLevel`	*I*	-	Controls the amount of information output.	0, 1, 2
`coupleOutlets`	*C*	-	Switches mapping of outlets. Requires the dictionary `mapOutletsToDetchemTubes` and the list `namesOfCoupledOutlets.`	on, off
Geometry				

directionOfPatches	*C*	-	Global direction of the wall patches coupled to DETCHEM tubes.	X, x, Y, y, Z, z
lengthOfDetchemTubes	*R*	m	Length of the coupled tubes.	any
numberOfDetchemBands	*I*	-	Number of bands used in DETCHEM calculations.	any
Boundary mapping				
numberOfDetchemTubes	*I*	-	Total number of tubes for which DETCHEM is called.	any

mapPatchesToDetchemTubes
{
namesOfReferencePatches
namesOfCoupledPatches
mapsToDetchemTubes
}
	Dict	-	Dictionary to control for which wall patches a DETCHEM calculation is performed. Requires the lists:
			namesOfReferencePatches
			namesOfCoupledPatches
			mapsToDetchemTubes.
				-

namesOfReferencePatches (Name_1 Name_2);	*List*	-	For each of the wall patches Name_x a DETCHEM calculation is performed. The number of entries in the list has to be the same as the integer numberOfDetchemTubes.	-
namesOfCoupledPatches (Name_1 Name_2);	*List*	-	For each of these wall patches with the name Name_x a DETCHEM calculation is applied. The number of entries in the list has to be the same as the number of entries in the list mapsToDetchemTube.	-
mapsToDetchemTubes (Number_1 Number_2);	*List*	-	For each of these wall patch names given in namesOfCoupledPatches, a tube number has to be allocated. For the equivalent wall patch, the result of a DETCHEM calculation is applied. The number of entries in the list has to be the same as the number of entries in the list namesOfCoupledPatches.	-

mapOutletsToDetchemTubes
{
namesOfCoupledOutlets
}
	Dict	-	Dictionary to control which outlets are used as inlets in the DETCHEMcalculations. Requires the list namesOfCoupledOutlets.	-

`namesOfCoupledOutlets ( Name_1 Name_2 );`	*List*	-	For each of the boundary outlets `Name_x` area-averaged values are calculated and used as inlet conditions by DETCHEM. If `coupleOutlets` is active, the number of entries in the list has to be same as the integer `numberOfDetchemTubes.`	-
Numerical parameters				
`repetitionFactor`	*I*	-	DETCHEM is called at every n^{th} iteration	any

18.3.5 Start of a Calculation with DUO

Start the OpenFOAM application **DUO** by typing the command ***chtMultiRegionSimpleFoamChemDuo***. All the keywords required for the coupling to DETCHEM (see table Tab. 4 1) have to be present in the file `<caseDir>/`***couplingParameters***. Now, the entry for the keyword `coupling` has to be:

`coupling on;` .

For the first run, it is advisable to activate additional output by setting

`debugLevel 2;` .

You could also create an executable shell script named ***start*** with the following entry:

```
#!/bin/csh
chtMultiRegionSimpleFoamChemDuo .
```

If you start the calculation with the command

`Start > log &` ,

the job will work in the background and the output will be written into the file ***log***.

18.3.6 Data Exchange

The external coupling between the two codes is done using ASCII data files. These files are written into the folder `system` of the OpenFOAM case directory `<caseDir>`. The procedure of data exchange between several directories might seem a little bit illogical at first sight. This is because the input and output from/to other directories than normally used from OpenFOAM (e.g. to the folder `detchem`) is complicated to handle. Therefore the exchange of data is managed within the folder `system`.

A: Intermediate Data from OpenFOAM to DETCHEM

For each iteration or time step, OpenFOAM creates two input files for every coupled tube. In the following, the symbol *** stands for the identification number of a tube. The temperatures at the outer walls of the coupled tubes are written to the file

`<caseDir>/system/`***walltemp-to-DETCHEM-****** .

The conditions at the outlets of the CFD mesh which shall be used as inlet conditions in the DETCHEM calculations are written to the file

`<caseDir>/system/`***outlet-to-DETCHEM-****** .

For every time step in which a DETCHEM program is called, these files are copied to the folder `<caseDir>/detchem` with a new name (cut of identification number). This is illustrated in the following, in which → represents the copying process:

system/walltemp-to-DETCHEM- → *detchem/walltemp-to-DETCHEM***
system/outlet-to-DETCHEM- → *detchem/outlet-to-DETCHEM***

The filenames ***walltemp-to-DETCHEM*** and ***outlet-to-DETCHEM*** are linked as includes in the files ***plug.inp*** and ***channel.inp*** and interpreted from the codes ***plug*** or ***channel***.

B: Intermediate Data from DETCHEM to OpenFOAM

For each calculation of a tube with the compiled codes ***plug*** or ***channel*** in the directory `detchem`, a file `<caseDir>/detchem/`***heatflux-from-DETCHEM*** containing the profile of the heatflux at the tube wall is created. Afterwards, this file is copied to the folder `<caseDir>/system` with a new name. The file name is ***heatflux-from-DETCHEM-*****, in which the symbol *** stands for the identification number of the tube. This is illustrated in the following, in which → represents the copying process:

$$\textit{detchem/heatflux-from-DETCHEM} \rightarrow \textit{system/heatflux-from-DETCHEM-***} \quad .$$

The heatfluxes in the files ***heatflux-from-DETCHEM-***** are red and mapped to the wall patches of the boundary handling subroutine *detchemHeatFluxTemperatureFvPatchScalarField.C*.

18.3.7 Data Output

A: Information Output

During the analysis of the keywords in the file ***couplingParameters***, the code checks the user parameters and stops if something is specified wrongly. Additionally, several quantities and variables are checked during the calculation. The code stops if something seems to be false. By setting the value of the keyword `debugLevel`, **DUO** produces additional output coming from OpenFOAM, which is always helpful for debugging. All output is displayed on the screen.

Additionally, DETCHEM produces screen output depending on the monitor option specified with the tag **<OUTPUT>** in the input files ***plug.inp*** or ***channel.inp***. With **monitor=1**, the screen output reports the iteration count and the temperature. With **monitor=2**, it reports the iteration count, temperature and the species mass fractions. With **monitor=no** no additional output is generated.

B: Solution Data from OpenFOAM

The `<time>` directories contain the result files. For an example case with the name reformer, the velocity field U for the timestep 10 s is stored in the file `run/reformer/10/`***U***. The solution files can be converted to the TECPLOT format using the utility ***foamToTecplot360***.

C: Solution Data from DETCHEM

Depending on the options set in the input file, DETCHEM$^{\text{PLUG}}$ or DETCHEM$^{\text{CHANNEL}}$ produces several file outputs. All the output files are in the TECPLOT format. In the following, the symbol *** stands for the value of the tag **Filenumber** in ***plug.inp*** or ***channel.inp***. The output files ***outg***.plt*** list velocity, residence times, temperature density and the gas-species mass or mole fractions, depending on whether the species tag in the **<OUTPUT>** element is supplied with mass or mole option. If the **coverage** option is chosen, the files ***outs***.plt*** will be produced, which list the surface coverages along the axial position of the reactor. If the **transfercoeff** option is chosen, the files ***trans***.plt*** will be produced and will list the mass and heat transfer coefficient calculated using the correlations. The file ***summary.dat*** contains the exit temperature and composition of the species listed in the **<SUMMARY>** tag. If the **<SUM>** tag is additionally used, the file ***summary.dat*** contains the sum of the mass/mole fraction of the species listed in the **<SUM>** tag.

18.4 Appendix

18.4.1 Example of a Keyword File *couplingParameters*

In the following, an example of a parameter file named ***couplingParameters*** is printed and described. The syntax is the same as in C++, because the file is handled as an include file. Lines with in the first column and between the symbols /* and */ are comments.

The sequence of the parameters and the positions of the dictionaries are arbitrary. The sequence of the lists in each dictionary and the sequence of entries inside a list are also in a random order.

```
/*---------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  1.7.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
|    \\/     M anipulation  |                                                 |
```

```
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      couplingParameters;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

//==================================
//STANDARD PARAMETERS
//==================================
coupling              plug;
coupleOutlets           on;
repetitionFactor        10;
debugLevel               1;
directionOfPatches       z;
numberOfDetchemTubes     2;
lengthOfDetchemTubes  0.01;
numberOfDetchemBands    50;

//==================================
//SUBDICTIONARY 1:
//MAPPING PARAMETERS FOR WALLS
//==================================
mapPatchesToDetchemTubes
{
   //-------------------------------------
   //WORD LIST
   //-------------------------------------
   namesOfReferencePatches
   (
   WALL_0202
   WALL_0204
   );

   //-------------------------------------
   //WORD LIST
   //-------------------------------------
   namesOfCoupledPatches
   (
   WALL_0201
   WALL_0202
   WALL_0203
   WALL_0204
   );

   //-------------------------------------
   //WORD LIST
   //-------------------------------------
   mapsToDetchemTubes
   (
   1
   1
   2
   2
   );
}

//==================================
//SUBDICTIONARY 2:
//MAPPING PARAMETERS FOR OUTLETS
//==================================
mapOutletsToDetchemTubes
{
   //-------------------------------------
   //WORD LIST
   //-------------------------------------
   namesOfCoupledOutlets
   (
   FLUID_OUT_0202
   FLUID_OUT_0204
   );
}

// ************************************************************************* //
```

File Header

After the several comment lines usually used in OpenFOAM files, a block starts, in which some information about the file can be found. In this part, the user does not have to change anything. Next, the standard parameters for the coupling can be found. Make sure that each line ends with the symbol "*;*". After that, the dictionaries including the parameters for the control of the patch mappings are shown.

Standard parameters

The coupling to DETCHEM itself is activated (`coupling on`). The coupling of outlets to DETCHEM inlets is active (`coupleOutlets on`). A calculation with **plug** or **channel** is performed at every 10th timestep (`repetitionFactor 10`) of OpenFOAM. The amount of information output is moderate (`debugLevel 1`). The tubes point into the direction z (`directionOfPatches z`). The count of tubes of the monolith for which a DETCHEM-calculation is performed is 2 (`numberOfDetchemTubes 2`) , The length of the tubes in DETCHEM and OpenFOAM is 0.01 m (`lengthOfDetchemTubes 0.01`). The number of bands or tube wall sections in the DETCHEM calculations is 50 (`numberOfDetchemBands 50`).

Dictionary mapPatchesToDetchemTubes

The word list `namesOfReferencePatches` includes the names of two wall patches. For each of these patches, a DETCHEM calculation of a tube is performed. This means, that for each of the tubes, the distribution of the wall temperature is taken from the corresponding wall patch of OpenFOAM. Caution: the count of names in the word list has to be the same as the value of the parameter `numberOfDetchemTubes`.

The word list `namesOfCoupledPatches` includes the names of four wall patches. For each of these patches, a result of a DETCHEM calculation is applied. For each of the tubes, the distribution of the wall temperature is taken from the corresponding wall patch of OpenFOAM. This means that the wall heat flux of a wall patch is set using a result from a DETCHEM calculation. For each of the coupled patches, a mapping number has to be allocated. Caution: each reference patch is also a coupled patch.

The word list `mapsToDetchemTubes` includes four mapping numbers. Each number belongs to one of the names of the coupled patches. The number represents the index of the tube calculated for which the resulting heat flux is mapped to the wall patch. Caution: each coupled patch has to have its own number. This means that the counts of elements in the list `namesOfCoupledPatches` and the list `mapsToDetchemTubes` have to be the same.

Again, the explanation of the wall mappings:

For the two wall patches `WALL_0202` and `WALL_0204`, two DETCHEM calculations for two tubes will be performed. For each of the calculations, the boundary temperature of the corresponding patch is used as a boundary condition. The result (boundary heat flux) of the first tube calculation is used for TWO boundary patches in OpenFOAM, for `WALL_0201` AND `WALL_0202`. Both patches have the same tube number “1” in the list `mapsToDetchemTubes`. The result of the second tube calculation is used for TWO boundary patches in OpenFOAM, for `WALL_0203` AND `WALL_0242`. Both patches have the same tube number “2” in the list `mapsToDetchemTubes`.

Dictionary mapOutletsToDetchemTubes

The word list `namesOfCoupledOutlets` includes the names of two outlet patches. For each of these patches, the outflow conditions (average velocity and averaged temperature) are calculated and used as input for the DETCHEM calculations. For the calculation of tube 1, the outflow conditions of the patch `FLUID_OUT_0202` are applied, for the calculation of tube 2 the outflow conditions of the patch `FLUID_OUT_0204`.

Boundary conditions

Every tube interchange surface has to be defined with the boundary condition name `detchemHeatFluxTemperature` in the file `<time>/SOLID/`$\boldsymbol{T}$. For the actual case the following entry would be necessary.

```
WALL_0201
    {
        type            detchemHeatFluxTemperature;
        K               K;
        q               uniform 0;
        gradient        uniform 0;
        value           uniform 0;
    }
WALL_0202
    {
        type            detchemHeatFluxTemperature;
        K               K;
        q               uniform 0;
        gradient        uniform 0;
        value           uniform 0;
    }
WALL_0203
    {
        type            detchemHeatFluxTemperature;
        K               K;
```

```
        q               uniform 0;
        gradient        uniform 0;
        value           uniform 0;
    }
WALL_0204
    {
        type            detchemHeatFluxTemperature;
        K               K;
        q               uniform 0;
        gradient        uniform 0;
        value           uniform 0;
    }
```

The values for the parameters *K*, *q*, *gradient* and *value* are arbitrary, because they are not used.

18.4.2 Example of an Input File *plug.inp*

In the following, an example of a parameter file named ***plug.inp***, which is necessary for **DUO** to call the executable DETCHEM$^{\text{PLUG}}$, is printed and described. This file has to be located in the folder `<caseDir>/detchem`.

```
{verbose n}
{include species.inp}
{include mech.inp}
{include walltemp-to-DETCHEM}
{include outlet-to-DETCHEM}

<SURFACE-MODEL>
  <CHEMSURF>
    hini=1.d-10
    time=10.
  </CHEMSURF>
  Fcatgeo=5
</SURFACE-MODEL>

<PFR>
 <GEOMETRY >
   reactor=cylindrical
  <SECTION>
    length =0.0002
    rin=0.000796
    rout=0.000796
    isothermal=n
    adiabatic=n
    mass-transfer=n
    axial-temp-profile=n
    gaschem=n
    surfchem=y
    tw={get T1}
  </SECTION>
...
    <SECTION>
    length =0.0002
    rin=0.000796
    rout=0.000796
    isothermal=n
    adiabatic=n
    mass-transfer=n
    axial-temp-profile=n
    gaschem=n
    surfchem=y
    tw={get T50}
  </SECTION>
</GEOMETRY>
  <INLET>
    u0={get Vinlet}
    p0=1.e+05
    T0={get Tinlet}
    <MOLEFRAC>
     CH4    0.133
     O2     0.0666
     N2     *
    </MOLEFRAC>
   </INLET>
  <SOLVER>
     ini_step=1e-16
     max-h = 2e-4
    <TOLERANCE>
        s_aTol=1.0d-06
        s_rTol=1.0d-05
        t_aTol=1.0d-06
        t_rTol=1.0d-06
        al_aTol=1.0d-06
        al_rTol=1.0d-06
     </TOLERANCE>
   </SOLVER>
```

```
     <OUTPUT>
        file_num={get Filenumber}
        title="CH4"
#       coverage = y
        species = mole
        monitor=no
     </OUTPUT>
</PFR>
```

Description

For the explanation of the keywords, the user should read the manual of DETCHEM^{PLUG}. Here, only the entries which are essential for the performance of **DUO** are explained.

An important fact is that we have a tube which is divided in 50 geometrical sections or bands. The definitions for each band can be found between the tags **<SECTION>** and **</SECTION>** (only two of them are shown in the example).

It should be recapitulated that the calculations for all coupled tubes (or channels) take place in the same directory using the same input file ***plug.inp***. The names of the include files are therefore always constant. But the content of the include files changes for each tube because it is overwritten by **DUO** for each DETCHEM calculation.

The included files are linked in the first lines. In the file ***walltemp-to-DETCHEM***, the wall temperatures for the bands and the tube identification number can be found. Here is an example of the content of the file:

```
{Define Filenumber 2}
{Define T1 1050.}
...
{Define T50 1000.}
```

In the file ***outlet-to-DETCHEM***, the inlet conditions for the tubes can be found. Here is an example of the content of the file:

```
{Define Tinlet 836.70239}
{Define Vinlet 0.54443955}.
```

The name **<name of variable>** of each variable (**T1**, **T2**, **Filenumber**, **Vinlet**, **Tinlet**) is referred to by using the tag {**get <name of variable>**} in ***plug.inp***. This means that the value of the variable is inserted here while running DETCHEM^{PLUG}. Therefore, the "real" content of the file ***plug.inp*** is different for every call.

Caution

If the coupling of the outlets is NOT activated in the parameter file ***couplingParameters*** (`coupleOutlets off`), the values for **u0** and **T0** have to be set by values. Additionally, the appropriate include has to be removed or to be commented (**#**{**include outlet-to-DETCHEM**}).

18.4.3 Example of an Input File *channel.inp*

In the following, an example of a parameter file named ***channel.inp***, which is necessary for **DUO** to call the executable DETCHEM^{CHANNEL}, is printed and described. This file has to be located in the folder `<caseDir>/-detchem`.

```
{verbose n}
{include species.inp}
{include mech.inp}
{include walltemp-to-DETCHEM}
{include outlet-to-DETCHEM}

<SURFACE-MODEL>
  <CHEMSURF>
   hini=1.d-10
   time=1.d0
  </CHEMSURF>
   Fcatgeo=185
  <MIXED_DIFF>
    <ZONE>
       ngrid=10
       aspect=1
       thickness=4.d-5
     </ZONE>
     porosity=0.5
```

```
     tau=3
     diameter=2.3e-8
     <SOLVER>
       time=10
       hini=1.d-4
       rtol=1.d-4
       atol=1.d-20
     </SOLVER>
  </MIXED_DIFF>
   EFF_MODEL = O2
</SURFACE-MODEL>

 <CHANNEL Version=2.0>
 <BASICS>
   title="CH4_auf_Rh"
   ngrid=18
   zmax=0.01
   rmax=0.0004384
 </BASICS>

 <SECTION>
   gaschem=n
   <WALL>
     surfchem=y
     mechanism="Rh"
   </WALL>
 </SECTION>

 <SOLVER>
   hini=1.d-10
   hmax=0.01
 </SOLVER>

 <OUTPUT>
   outg=y
   outs=y
   outflux=y
   summary=n
   file={get Filenumber}
   copy=n
   monitor=1
   molefrac=y
 </OUTPUT>

<INLET>
  u0={get Vinlet}
  p =1.e+05
  T0={get Tinlet}
  <MOLEFRAC>
  # C/O: 1.0
  CH4  0.13
  O2  0.0666
  N2  *
</MOLEFRAC>
</INLET>

<TPROFILE>
{include points.txt}
</TPROFILE>
</CHANNEL>
```

The file ***points.txt*** includes the temperatures of the 50 bands or sections which were choosen here to exchange data between the codes.

Caution: DETCHEM$^{\text{CHANNEL}}$ needs here 51 data points, because the temperature is specified at the boundaries of the sections. The temperatures calculated in OpenFOAM are the average temperatures of each band. Therefore we get 50 values. We use the temperature of band 50 for the left and the right boundary of the last section. Contrary, in DETCHEM$^{\text{PLUG}}$, the temperature is defined for the whole band.

```
<DISCRETE>
0.0 {get T1}
0.0002 {get T2}
0.0004 {get T3}
....
0.0096 {get T49}
0.0098 {get T50}
0.01 {get T50}
</DISCRETE>
```

Description

For the explanation of the keywords, the user should read the manual of DETCHEM$^{\text{CHANNEL}}$. Here, only the entries which are essential for the performance of **DUO** are explained.

In contrast to DETCHEM^{PLUG} we need to define only one section. The number of bands or sections where we want to define a temperature is defined in between the keywords **<TPROFILE>** and **</TPROFILE>**.

It should be recapitulated that the calculations for all coupled tubes (or channels) take place in the same directory using the same input file ***channel.inp*** The names of the include files are therefore always constant. But the content of the include files changes for each tube because it is overwritten by **DUO** for each DETCHEM^{CHANNEL} calculation.

The included files are linked in the first lines. In the file ***walltemp-to-DETCHEM***, the wall temperatures for the bands and the tube identification number can be found. Here is an example of the content of the file:

```
{Define Filenumber 2}
{Define T1 1050.}
...
{Define T50 1000.}
```

In the file ***outlet-to-DETCHEM***, the inlet conditions for the tubes can be found. Here is an example of the content of the file:

```
{Define Tinlet 836.70239}
{Define Vinlet 0.54443955}.
```

The name **<name of variable>** of each variable (**T1**, **T2**, **Filenumber**, **Vinlet**, **Tinlet**) is referred to by using the tag {**get <name of variable>**} in ***channel.inp***. This means that the value of the variable is inserted here while running DETCHEM^{CHANNEL}. Therefore, the "real" content of the file ***channel.inp*** is different for every call.

Caution

If the coupling of the outlets is NOT activated in the parameter file ***couplingParameters*** (`coupleOutlets off`), the values for **u0** and **T0** have to be set by values. Additionally, the appropriate include has to be removed or to be commented (**#**{**include outlet-to-DETCHEM**}).

Chapter 19

DETCHEM Parameters

Up to version 2.1 DETCHEM was mainly written in Fortran 77, which did not offer dynamic memory allocation. In order to give more flexibility to the user, version 2.2 is based on Fortran 90 and can no longer be compiled using a Fortran 77 compiler. The main parameters (e. g. number of species and reactions) can now be changed during run-time. Recompilation is no longer necessary.

The default size parameters set in the distribution of DETCHEM may be not sufficient to meet the needs of the user. If you try to define a case for that more memory allocation is necessary, the program will usually abort with an error message. (In some rare cases an exception is not found – the execution may then terminate surprisingly or give wrong results.)

```
 ERROR: More than  50 cells defined
 increase parameter MGRmaxX in <PARAMETERS> section
```

If you receive such an error message, first check your input file for wrong parameters. If you do not know which option may have caused this error message, use the {**verbose yes**} command in the input file. In the **<SPECIES>**, **<MECHANISM>**, **<CHANNEL>**, and **<MONOLITH>** sections, you can redefine the parameters directly in the input file. For this, insert a **<PARAMETERS>** tag directly after the opening tag of one of these sections. (The **<PARAMETERS>** tag must be the first evaluatable option in these sections.) For instance:

```
<SPECIES>
  <PARAMTERS>
    DCSGMAX = 500   # maximum number of gas-phase species
    DCSMMAX = 50    # maximum number of collision efficiencies
  </PARAMETERS>
  <GASPHASE>
    ...             # list of species
  </GASPHASE>
</SPECIES>

<MECHANISM>
  <PARAMETERS>
     DCRGMAX = 2000 # maximum number of gas-phase reactions
     DCRGTMAX=  100 # maximum number of Troe reactions
     DCRGGMAX=   20 # maximum number of global gas-phase reactions
  </PARAMETERS>
  <GASPHASE>
    ...             # gas-phase mechanism
  </GASPHASE>
</MECHANISM>
...
```

In case dynamic memory allocation is not yet available, you have to change the parameter in the file named by the error message and recompile the application. If you change a parameter in on of the DETCHEM libraries, it is required to recompile the whole package by calling **make** in the DETCHEM root directory.

The following table gives an overview of the parameter files

module	filename	kind of parameters
lib_input	***constINP.F***	line and name length, include depth
	commINPlink.F	tag depth, number of options in one tag
lib_detchem	***parameterDC.f90***	number of species, reactions, and surface types (fully dynamic)
lib_washcoat	***parameterWC.f90***	number of washcoat grid points
CHANNEL	***parameterBL.f90***	number of radial grid points, number of axial sections (fully dynamic)
GRIDGEN3D	***commGRID.F***	number of points, cells, and layers
MONOLITH	***parameterM.f90***	grid size, number of materials and boundaries, output times, channel types (fully dynamic)

Chapter 20

Installation guide

The following section describes how to install DETCHEM

20.1 Extract data

DETCHEM is distributed as a tar (***.tar***) or zipped tar (***.tgz***) file. Extract the tar file in your desired directory by

```
tar -xvf detchem.tar
```

or the compressed file by

```
tar -zxvf detchem.tgz
```

Now you will find the following directory structure:

- **code**
  - **lib_input**
  - **lib_utils**
  - **lib_detchem**
  - **lib_washcoat**
  - **CHEMINP**
  - **GASPROBE**
  - **...**
  - **MONOLITH**
  - ***Makefile***
  - ***Makefile.common***
- **examples**
  - **CHEMINP**
  - **GASPROBE**
  - **...**
  - **MONOLITH**
- **bin**
- ***README***

20.2 Run Make

Each of code subdirectories contains a ***Makefile***. System specific information that need to be known to all ***Makefile***s is saved in ***Makefile.common***. Define the commands to call the compiler and linker and their flags in this file. On some systems, relative path names in Makefiles seem not to work properly. In this case also define the absolute path to your DETCHEM directory in ***Makefile.common***. An example file is show below.

```
# Compiler Options for DETCHEM package

# DETCHEM_PATH: absolute path to current directory
#               assign a value if you want make to use asolute path names
#               when left blank relative paths will be used
DETCHEM_PATH =

# fortran compiler
FC       = g95
# compiler flags
FCFLAGS  = -c -O2
# linker
LD       = g95
# linker flags
LDFLAGS  =


# parallel version

# fortran compiler
PFC      = g95
# compiler flags
PFCFLAGS = $(FCFLAGS)
# linker
PLD      = g95
# linker flags
PLDFLAGS = $(LDFLAGS)
# MPI library
PLIBS    = -lmpi
```

You can call the ***Makefile*** in each code subdirectory seperately. More convenient is to call

`make`

just once in the **`code`** directory. It automatically calls **`make`** in all subdirectories in the required order. For generating the parallel version of DETCHEM$^{\text{MONOLITH}}$ type

`make parallel`

The executabels of all modules will be saved in the **`bin`** directory. (A **`bin`** directory will be created automatically by **`make`** if not present) In case you changed some code (e. g. user-defined functions), call **`make`** to update the executables. When you have changed size parameters for memory allocation, it is strongly suggested to remove all compiled files by typing

`make clean`

before recompiling the code again.

Installing DC4FLUENT

See chapter **??** for installation guide lines of **DC4FLUENT**.

List of Scientific Publications Using DETCHEM

1. O. Deutschmann and L. D. Schmidt. Two-Dimensional Modeling of Partial Oxidation of Methane on Rhodium in a Short Contact Time Reactor. Proc. Comb. Inst. 27 (1998), 2283-2291

2. L. D. Schmidt, O. Deutschmann, and C. T. Goralski, Jr. Modeling the Partial Oxidation of Methane to Syngas at Millisecond Contact Times. Natural Gas Conversion V, Studies in Surface Science and Catalysis 119, p. 685-692, Elsevier, Amsterdam, 1998.

3. O. Deutschmann and L. D. Schmidt. Modeling the Partial Oxidation of Methane in a Short Contact Time Reactor. AIChE J. 44 (1998) 2465-2476.

4. O. Deutschmann, L. D. Schmidt, J. Warnatz. Simulation of Reactive Flow in a Partial Oxidation Reactor with Detailed Gas Phase and Surface Chemistry Models, In: Scientific Computing in Chemical Engineering II. Computational Fluid Dynamics, Reaction Engineering, and Molecular Properties. F. Keil, W. Mackens, H. Voß, J. Werther (Eds.). p. 368-375, Springer, 1999.

5. L. L. Raja, R. J. Kee, O. Deutschmann, J. Warnatz, L. D. Schmidt. A Critical Evaluation of Navier-Stokes, Boundary-Layer, and Plug-Flow Models for the Simulation of Flow and Chemistry in a Catalytic Combustion Honeycomb Channel. Catalysis Today 59 (2000) 47-60.

6. O. Deutschmann, L. Maier, U. Riedel, A. H. Stroemann, R. W. Dibble. Hydrogen Asssisted Catalytic Combustion of Methane on Platinum. Catalysis Today 59 (2000) 141-150.

7. J. Braun, T. Hauber, H. Tbben, P. Zacke, D. Chatterjee, O. Deutschmann, J. Warnatz. Influence of Physical and Chemical Parameters on the Conversion Rate of a Catalytic Converter: A Numerical Simulation Study. SAE paper 2000-01-0211 (2000)

8. D. K. Zerkle, M. D. Allendorf, M. Wolf, O. Deutschmann. Modeling of On-Line Catalyst Addition Effects in a Short Contact Time Reactor. Proc. Combust. Inst. 28 (2000) 1365- 1372.

9. D.K. Zerkle, M.D. Allendorf, M. Wolf, O. Deutschmann. Understanding Homogeneous and Heterogeneous Contributions to the Platinum-Catalyzed Partial Oxidation of Ethane in a Short Contact Time Reactor. J. Catal. 196 (2000) 18-39

10. O. Deutschmann, R. Schwiedernoch, L.I. Maier, D. Chatterjee. Natural Gas Conversion in Monolithic Catalysts: Interaction of Chemical Reactions and Transport Phenomena. Natural Gas Conversion VI, Studies in Surface Science and Catalysis 136, E. Iglesia, J.J. Spivey, T.H. Fleisch (eds.), p. 215-258, Elsevier, 2001

11. J. M. Redenius, L. D. Schmidt, O. Deutschmann. Millisecond Catalytic Wall Reactors: I. Radiant Burner. AIChE J. 47(2001) 1177-1184

12. S. Tischer, C. Correa, O. Deutschmann. Transient three-dimensional simulation of a catalytic combustion monolith using detailed models for heterogeneous and homogeneous reactions and transport phenomena. Catalysis Today 69 (2001) 57-62

13. D. Chatterjee, O. Deutschmann, J. Warnatz. Detailed surface reaction mechanism in a three-way catalyst. Faraday Discussions 119 (2001) 371-384

14. J. Braun, T. Hauber, H. Tbben, J. Windmann, P. Zacke, D. Chatterjee, C. Correa, O. Deutschmann, L. Maier, S. Tischer, J. Warnatz. Three-Dimensional Simulation of the Transient Behavior of a Three-Way Catalytic Converter. SAE Technical paper 2002-01- 0065 (2002)

15. R. P. O Connor, L.D. Schmidt, O. Deutschmann. Simulating the Millisecond Oxidation of Cyclohexane by Coupled Chemistry and Fluid Dynamics. AIChE J. 48 (2002), 1241-1256

16. R. Schwiedernoch, S. Tischer, C. Correa, O. Deutschmann, J. Warnatz. Experimental and numerical investigation of the ignition of methane combustion in a platinum-coated honeycomb monolith. Proc. Combust. Inst. 29 (2002)

17. K. Maruta, K. Takeda, J. Ahn, K. Borer, L. Sitzki, P. D. Ronney, O. Deutschmann. Extinction Limits of Catalytic Combustion in Microchannels. Proc. Combust. Inst. 29 (2002).

18. R. Schwiedernoch, S. Tischer, C. Correa, O. Deutschmann. Experimantal and Numerical Study of the Transient Behaviour of a Catalytic Partial Oxidation Monolith. Chem. Eng. Sci. 58 (2003), 633-642.

19. J. Windmann, J. Braun, P. Zacke, S. Tischer, D. Chatterjee, O. Deutschmann, J. Warnatz. Impact of the Inlet Flow Distribution on the Light-Off Behaviour of a 3-Way Catalytic Converter. SAE Technical Paper 2003-01-0937 (2003).

20. R. Schwiedernoch, S. Tischer, H.-R. Volpp, O. Deutschmann. Towards a better understanding of transient processes in catalytic oxidation reactors. Natural Gas Conversion VI, Studies in Surface Science and Catalysis 147 (2004), 511-516.

21. M. Bizzi, R. Schwiedernoch, O. Deutschmann, G. Saracco. Modeling the Partial Oxidation of Methane in a Fixed Bed with Detailed Chemistry. AIChE J. 50 (2004), 1289 - 1299.

22. S. Tischer, O. Deutschmann. Recent advances in numerical modeling of catalytic monolith reactors. Catalysis Today (2005) in press.

23. V. M. Janardhanan, O. Deutschmann. CFD analysis of a solid oxide fuel cell with internal reforming: Coupled interactions of transport, heterogeneous catalysis and electrochemical processes. J. Power. Sources. 162 (2006), 1192-1202.

24. V. M. Janardhanan, O. Deutschmann. Numerical study of mass and heat transport in solid oxide fuel cells running on humidified methane. Chem. Eng. Sci. (in press)

25. V. M. Janardhanan, V. Heuveline, O. Deutschmann. Performance evaluation of planar solid oxide fuel cells. J. Power. Sources. (accepted).